Fitting Tree Metrics and Ultrametrics in Data Streams

Amir Carmel[*]   Debarati Das[†]   Evangelos Kipouridis [‡]   Evangelos Pipis[§]

Abstract

Fitting distances to tree metrics and ultrametrics are two widely used methods in hierarchical clustering, primarily explored within the context of numerical taxonomy. Formally, given a positive distance function $D : \binom{V}{2} \to \mathbb{R}_{>0}$, the goal is to find a tree (or an ultrametric) T including all elements of set V, such that the difference between the distances among vertices in T and those specified by D is minimized. Numerical taxonomy was first introduced by Sneath and Sokal [Nature 1962], and since then it has been studied extensively in both biology and computer science.

In this paper, we initiate the study of ultrametric and tree metric fitting problems in the semi-streaming model, where the distances between pairs of elements from V (with $|V| = n$), defined by the function D, can arrive in an arbitrary order. We study these problems under various distance norms; namely the ℓ_0 objective, which aims to minimize the number of modified entries in D to fit a tree-metric or an ultrametric; the ℓ_1 objective, which seeks to minimize the total sum of distance errors across all pairs of points in V; and the ℓ_∞ objective, which focuses on minimizing the maximum error incurred by any entries in D.

- Our first result addresses the ℓ_0 objective. We provide a single-pass polynomial-time $\tilde{O}(n)$-space $O(1)$ approximation algorithm for ultrametrics and prove that no single-pass exact algorithm exists, even with exponential time.

- Next, we show that the algorithm for ℓ_0 implies an $O(\Delta/\delta)$ approximation for the ℓ_1 objective, where Δ is the maximum, and δ is the minimum absolute difference between distances in the input. This bound matches the best-known approximation for the RAM model using a combinatorial algorithm when $\Delta/\delta = O(n)$.

- For the ℓ_∞ objective, we provide a complete characterization of the ultrametric fitting problem. First, we present a single-pass polynomial-time $\tilde{O}(n)$-space 2-approximation algorithm and show that no better than 2-approximation is possible, even with exponential time. Furthermore, we show that with an additional pass, it is possible to achieve a polynomial-time exact algorithm for ultrametrics.

- Finally, we extend all these results to tree metrics by using only one additional pass through the stream and without asymptotically increasing the approximation factor.

---

[*]Pennsylvania State University, United States. Part of this work was done while the author was affiliated at Weizmann Institute of Science, Israel. `amir6423@gmail.com`

[†]Pennsylvania State University, United States. `debaratix710@gmail.com`

[‡]Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany. `kipouridis@mpi-inf.mpg.de`

[§]National Technical University of Athens, Greece, and Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany. `evpipis@gmail.com`

1 Introduction

Hierarchical clustering is a method of cluster analysis that builds a hierarchy of clusters by starting with each data point as its own cluster and successively merging the two closest clusters until all points are merged into a single cluster or a stopping criterion is met. This method involves creating a *dendrogram*, a tree-like diagram, that records the sequence of merges or splits, allowing for easy visualization and interpretation of the hierarchical structure. Hierarchical clustering uses various distance metrics (e.g., Euclidean, Manhattan, cosine) and linkage criteria (e.g., single, complete, average, Ward's method), providing flexibility to tailor the analysis to specific data characteristics and clustering goals. It is versatile across different types of data, including numerical, categorical, and binary data, and has become the preferred method for analyzing gene expression data [D'h05] and constructing phylogenetic trees [AC11, KLNHM17]. Consequently, hierarchical clustering plays a significant role in both theory and practice across various domains, such as image processing to group similar regions within images [LC05], social network analysis to identify communities within a network [BBA75], and business and marketing to segment customers based on behavior, preferences, or purchasing patterns [KSVK20].

Tree metrics and ultrametrics are fundamental measures used in hierarchical clustering, where the distance between any two points is defined by the cost of the unique path connecting them in a tree-like structure. Formally, given a distance function $D : \binom{V}{2} \to \mathbb{R}_{>0}$, the goal is to find a tree T with positive edge weights, encompassing all elements of set V as vertices. This tree T should best match the distances specified by D. In the case of ultrametrics, the tree must be rooted, and all elements of V must appear as leaf nodes at the same depth.

The task of fitting distances with tree metrics and ultrametrics, often referred to as the numerical taxonomy problem, has been a subject of interest since the 1960s [CSE67, SS62, SS63]. One of the pioneering works in this area was presented by Cavalli-Sforza and Edwards in 1967 [CSE67]. Different formulations of the optimal fit for a given distance function D lead to various objectives, such as minimizing the number of disagreements (using the ℓ_0 norm of the error vector), minimizing the sum of differences (using the ℓ_1 norm), and minimizing the maximum error (using the ℓ_∞ norm).

Deploying hierarchical clustering (HC) algorithms faces significant challenges due to scalability issues, particularly with the rise of data-intensive applications and evolving datasets. As data volumes continue to grow, there is an urgent need for efficient algorithms tailored for large-scale models such as streaming, local algorithms, MPC, and dynamic models, given the large input sizes relative to available resources. In this work we study hierarchical clustering, focusing on tree metrics and ultrametrics in the semi-streaming model. The model supports incremental updates, keeping the information about the clusters current without the need to reprocess the entire dataset. This adaptability makes hierarchical clustering highly valuable for applications such as network monitoring and social media analysis, where real-time insights are essential [RGP08, LLM14, LL09].

A recent result [ACL$^+$22] studied hierarchical clustering (HC) in the graph streaming model, providing a polynomial-time, single-pass $\tilde{O}(n)$ space algorithm that achieves an $O(\sqrt{\log n})$ approximation for HC. When space is more limited, specifically $n^{1-o(1)}$, the authors show that no algorithm can estimate the value of the optimal hierarchical tree within an $o(\log n \log \log n)$ factor, even with $poly \log n$ passes over the input and exponential time.

A special case of the ultrametric fitting problem is where the tree depth is two, known as the *Correlation Clustering problem*. In this problem given a complete graph $G = (V, E)$ with edges labeled either similar (0) or dissimilar (1), the objective is to partition the vertices V into clusters to minimize the disagreements. After a decade of extensive research on correlation clustering in the semi-streaming setting [CDK14, ACG$^+$21, CLM$^+$21, AW22, BCMT22, CLMP22, BCMT23, CLP$^+$24], a recent breakthrough in [CLP$^+$24] introduces a single-pass algorithm that achieves a 1.847 approximation using $\tilde{O}(n)$ space. This directly improves upon two independent works [CKL$^+$24, MC23], both presenting single-pass algorithms achieving a $(3 + \varepsilon)$-approximation using $O(n/\varepsilon)$ space.

However, our understanding of streaming algorithms for larger depths, particularly within the context of ultrametrics and tree metrics, is very limited. The challenge arises from the fact that, unlike correlation clustering, which deals with only two distinct input distances, this problem may involve up to n^2 different distances, especially in a highly noisy input. Although the optimal output tree can be defined using at most n of these n^2 distances, identifying these n distances is non-trivial. As a result, in the worst case, it

may be necessary to store all observed input distances, which would require quadratic space if done naively. Additionally, the hierarchical nature of clusters at various tree depths introduces inherent dependencies among clusters at different levels. This complexity makes it highly challenging to adapt the ideas used in streaming algorithms for correlation clustering to ultrametrics and tree metrics. In this paper, we offer the first theoretical guarantees by providing several algorithmic results for fitting distances using ultrametrics and tree metrics in the semi-streaming setting under various distance norms.

1.1 Other Related Works

Ultrametrics and Tree metrics. The numerical taxonomy problem, which involves fitting distances with tree metrics and ultrametrics, was first introduced by Cavalli-Sforza and Edwards in 1967 [CSE67]. Day demonstrated that this problem is NP-hard for both ℓ_1 and ℓ_2 norms in the context of tree metrics and ultrametrics [Day87]. Moreover, these problems are APX-hard [CGW05], as inferred from the APX-hardness of Correlation Clustering, which rules out the possibility of a polynomial-time approximation scheme. On the algorithmic side, Harp, Kannan, and McGregor [HKM05] developed an $O(\min\{n, k \log n\}^{1/p})$ approximation for the ℓ_p objective in the ultrametric fitting problem, where k is the number of distinct distances in the input. Ailon and Charikar [AC11] improved this to an $O(((\log n)(\log \log n))^{1/p})$ approximation, which they extended to the tree metric using a reduction from Agarwala [ABF+99]. A recent breakthrough in [CDK+21] presented the first constant-factor approximation for the ℓ_1 objective for both ultrametric and tree metric.

The ℓ_0 objective was first investigated in [CFLDM22], which developed a novel constant-factor approximation algorithm. Charikar and Gao [CG24] improved the approximation guarantee to 5. For the weighted ultrametric violation distance, where the weights satisfy the triangle inequality, they provided an $O(\min\{L, \log n\})$ approximation, with L being the number of distinct values in the input. Kipouridis [Kip23] further extended these results to tree metrics.

Research into the ℓ_∞ numerical taxonomy began in the early 1990s. It was discovered by several authors that the ℓ_∞ case of the ultrametric fitting problem is solvable in polynomial time (in fact linear time in the input) and it is the only case with that property, whereas the problem of ℓ_∞ tree fitting is APX-hard [Kři88, CF00, FKW93, ABF+99, War92]. Since then, the ℓ_∞ Best-Fit Ultrametrics/Tree-Metrics problems were extensively studied from both mathematical and computational perspectives [CF00, BL17, Ber20, Ard05, DHH+05, MWZ99, CKL20, CDL21].

Correlation Clustering. The classic correlation clustering problem, introduced by Bansal, Blum, and Chawla [BBC02], can be visualized as a special case of ultrametrics where the tree's depth is bounded by two. Correlation clustering serves as a fundamental building block for constructing ultrametrics and tree metrics. Despite being APX-hard [CGW05], extensive research [BBC02, CGW05, CMSY15, CLN22, CLLN23] has aimed at finding efficient approximation algorithms, with the latest being a 1.437-approximation [CCL+24]. Correlation clustering also boasts a rich body of literature and has been extensively studied across various models designed for large datasets, including streaming [ACG+21, AW22, BCMT22, CLN22], MPC [CLM+21], MapReduce [CDK14], and dynamic models [BDH+19, BCC+24, DMM24].

Metric Violation Distance. Another counterpart of the ultrametric violation distance problem is the metric violation distance problem, which requires embedding an arbitrary distance matrix into a metric space while minimizing the ℓ_0 objective. While only a hardness of approximation of 2 is known, [GJ17, FRB18, GGR+20] provided algorithms with an approximation ratio of $O(OPT^{1/3})$. An exponential improvement in the approximation guarantee to $O(\log n)$ was achieved in [CFLDM22]. The maximization version of this problem is also well-motivated by its application in designing metric filtration schemes for analyzing chromosome structures, as studied in [DPS+13].

1.2 Our Contributions

In this work, we examine the problem of fitting tree metrics and ultrametrics in the semi-streaming model, focusing on the ℓ_0 and ℓ_∞ objectives. Note that storing the tree alone requires $\Omega(n)$ word space. Since we are working within the semi-streaming model, where $\tilde{O}(n)$ space is permitted, this is a natural consideration. Our results apply to the most general semi-streaming settings where the entries of the input distance matrix,

of size n^2, arrive one by one in some arbitrary order, possibly adversarially. Notably, all our algorithms require either one or two passes over the data while achieving constant factor approximations in polynomial time. Before discussing the key contributions of this work, we provide a formal definition of the problem.

Problem: ℓ_p Best-Fit Ultrametrics/Tree-Metrics

Input: A set V and a distance matrix $D : \binom{V}{2} \to \mathbb{R}_{>0}$.

Desired Output: An ultrametric (resp. tree metric) T that spans V and fits D in the sense of minimizing:

$$\|T - D\|_p = \sqrt[p]{\sum_{uv \in \binom{V}{2}} |T(uv) - D(uv)|^p}$$

For $p = 0$, the aim is to minimize the total number of errors. In other words, each pair comes with a request regarding their distance in the output tree, and our goal is to construct a tree that satisfies as many of these requests as possible, minimizing the total number of pairs whose distances are altered. This fundamental problem for ultrametrics, also known as the *Ultrametric violation distance* problem, was first investigated in [CFLDM22], in which a novel constant-factor approximation algorithm in the RAM model was developed. Charikar and Gao [CG24] further improved the approximation guarantee to 5.

We present for this problem a single-pass algorithm, in the semi-streaming setting, that provides a constant approximation and succeeds with high probability. We remark that straightforwardly adapting this algorithm in the RAM model yields a near-linear time algorithm ($\widetilde{O}(n^2)$, while the input size is $\Theta(n^2)$), improving over the best known $\Omega(n^4)$ time from [CFLDM22][1].

Theorem 1. *There exists a single pass polynomial time semi-streaming algorithm that w.h.p. $O(1)$-approximates the ℓ_0 Best-Fit Ultrametrics problem.*

Following, we show that this result also implies an approximation for the ℓ_1 objective.

Corollary 2. *Let δ (resp. Δ) be the smallest (resp. largest) absolute difference between distinct distances in D, for an ℓ_1 Best-Fit Ultrametrics instance. There exists a single pass polynomial time semi-streaming algorithm that w.h.p $O(\Delta/\delta)$-approximates the ℓ_1 Best-Fit Ultrametrics problem.*

Proof. Let T_0 (resp. T_1) be an ultrametric minimizing $\|T_0 - D\|_0$ (resp. $\|T_1 - D\|_1$), and T be the output from Theorem 1 (thus $\|T - D\|_0 = O(\|T_0 - D\|_0)$.

It holds that $\|T_1 - D\|_1 \geqslant \|T_1 - D\|_0 \delta$, as in the ℓ_1 objective we pay at least δ for each pair having a disagreement. Similarly $\|T - D\|_1 \leqslant \|T - D\|_0 \Delta = O(\|T_0 - D\|_0 \Delta) = O(\|T_1 - D\|_0 \Delta)$, by definition of T_0. $\square$

It is interesting to note that for the ℓ_1 objective, most recent approximation algorithms in the offline setting are not combinatorial, making it a significant challenge to adapt them to the semi-streaming model. The best known combinatorial approximation for ℓ_1 Best-Fit Ultrametrics and Tree-Metrics is $O(\Delta/\delta)$, when $\Delta/\delta = O(n)$ [AC11, HKM05], achieved through the so-called pivoting algorithm [2]. Unfortunately, this algorithm is very challenging to adapt to a single-pass semi-streaming setting as it generalizes the PIVOT-based algorithm of Correlation Clustering, which, despite extensive research, has not been adapted to semi-streaming settings with only a single pass without significant modifications [BCMT22, BCMT23, CKL+24, MC23]. Surprisingly, our $O(\Delta/\delta)$ approximation for the ℓ_1 objective is derived directly from the algorithm for the ℓ_0 objective, eliminating the need to explicitly use this pivoting approach.

Further, we contrast Theorem 1 by ruling out the possibility of a single-round exact algorithm, even with sub-quadratic space and exponential time. For this, we provide a new lower bound result for the correlation clustering problem, showing that any single-pass streaming algorithm with sub-quadratic space cannot output the optimal clustering nor can maintain its cost.

Theorem 3. *Any randomized single pass streaming algorithm that with probability greater than $\frac{2}{3}$ either solves the correlation clustering problem or maintains the cost of an optimal correlation clustering solution requires $\Omega(n^2)$ bits.*

---

[1] In [CFLDM22] the exact running time of the algorithm has not been analyzed, but there exist inputs where it must perform $\Omega(n^2)$ repetitions of a flat-clustering algorithm that takes $\Omega(n^2)$ time per repetition.

[2] The authors of [AC11] claim the approximation is proportional to the number of distinct distances. That is because of a simplification they make in the paper, that the distances are all consecutive positive integers starting from 1. For an example showing the $O(\Delta/\delta)$ analysis is tight, take $V = \{u_1, u_2, u_3\}$ and $D(uv) > \Delta, D(uw) = D(uv) - \delta, D(vw) = D(uv) - \Delta$. With probability $1/3$ we pick u as the pivot, and pay Δ, while the optimum solution only pays δ.

We then extend this result to ℓ_p Best-Fit Ultrametrics problems for $p \in \{0,1\}$, using the fact that correlation clustering is a special case of these problems (see e.g. [AC11]).

Corollary 4. *For $p \in \{0,1\}$, any randomized single pass streaming algorithm that with probability greater than $\frac{2}{3}$ either solves ℓ_p Best-Fit Ultrametrics or just outputs the error of an optimal ultrametric solution requires $\Omega(n^2)$ bits.*

Next, we consider the ℓ_∞ objective, where the goal is to minimize the maximum error. In Section 4 we provide a complete characterization of ℓ_∞ Best-Fit Ultrametrics in the semi-streaming model. We give a single pass algorithm with 2-approximation factor to this problem.

Theorem 5. *There exists a single pass polynomial time semi-streaming algorithm that 2-approximates the ℓ_∞ Best-Fit Ultrametrics problem.*

We contrast Theorem 5 by showing that this is the best approximation factor achievable using a single pass, even with sub-quadratic space and exponential time.

Theorem 6. *Any randomized one-pass streaming algorithm for ℓ_∞ Best-Fit Ultrametrics with an approximation factor strictly less than 2 and a success probability greater than $\frac{2}{3}$ requires $\Omega(n^2)$ bits of space.*

Moreover, we demonstrate that allowing two passes is sufficient for an exact solution. Therefore, we provide optimal tradeoffs between the number of passes and the approximation factor in all scenarios.

Theorem 7. *There exists a two-pass polynomial time semi-streaming algorithm that computes an exact solution to the ℓ_∞ Best-Fit Ultrametrics problem.*

In Section 5 we show that all aforementioned algorithms can be extended to tree metrics. This is achieved by providing reductions to the corresponding ultrametrics problems, requiring only one additional pass over the stream. The reductions used for the ℓ_0 and ℓ_∞ objectives differ significantly from each other.

Theorem 8. *There exists a two-pass polynomial time semi-streaming algorithm that w.h.p $O(1)$-approximates the ℓ_0 Best-Fit Tree-Metrics problem.*

Using the same arguments as in Corollary 2, we obtain an analogous result for the ℓ_1 objective.

Corollary 9. *Let δ (resp. Δ) be the smallest (resp. largest) absolute difference between distinct distances in D, for an ℓ_1 Best-Fit Tree-Metrics instance. There exists a two-pass pass polynomial time semi-streaming algorithm that w.h.p $O(\Delta/\delta)$-approximates ℓ_1 Best-Fit Tree-Metrics problem.*

Theorem 10. *There exists a two-pass polynomial time semi-streaming algorithm that 6-approximates the ℓ_∞ Best-Fit Tree-Metrics problem.*

1.3 Technique Overview

We provide a technical overview of the most technically novel contribution of our work, namely the results regarding the ℓ_0 Best-Fit Ultrametrics algorithm in the semi-streaming model (more details in Section 3).

1.3.1 Why previous ℓ_0 approaches cannot be adapted

In general, it is difficult to ensure a hierarchical structure while providing non-trivial approximation guarantees. In Hierarchical Clustering research, such results usually rely on one of two standard approaches, namely the top-down (divisive) approach, and the bottom-up (agglomerative) approach. In fact, with the exception of [CDK+21], results for ℓ_p Best-Fit Ultrametrics ($p < \infty$) [HKM05, AC11, CDK+21, CFLDM22, CG24] all rely on the divisive approach.

Non-divisive approaches. The only relevant result applying a non-divisive approach is that of [CDK+21], which crucially relies on a large LP. Unfortunately, it is not known how to solve such an LP in streaming.

Divisive approaches. A divisive algorithm starts with the root node (containing the whole V), computes its children (subsets V' at height h) based on some division strategy, and recurses on its children. Different division strategies have been employed, with the most prominent ones using the solution to an LP, or attempting to satisfy a particular (usually randomly chosen) element called the pivot, or solving some flat clustering problem. In what follows, we discuss why existing division strategies do not work in our case.

Correlation Clustering. Perhaps the most straightforward approach is to solve a (flat) clustering problem; for each pair of vertices $u, v \in V'$, we ideally want them together if $h > D(uv)$, and apart otherwise. This corresponds to the Correlation Clustering problem, which returns a clustering violating as few of our preferences as possible. Unfortunately, this approach does not work for ℓ_0 Best-Fit Ultrametrics, as certain choices that appear good locally (on a particular height h) may be catastrophic globally.

The first result for ℓ_0 Best-Fit Ultrametrics. The authors of [CFLDM22] overcame the shortcomings of Correlation Clustering as a division strategy by solving a particular flavor of it, called Agreement Correlation Clustering. This guaranteed further structural properties[3] that could be leveraged to provide $O(1)$ approximation for ℓ_0 Best-Fit Ultrametrics. However this approach is too strong to guarrantee in streaming, since one can recover adjacency information with black-box calls to an Agreement Correlation Clustering subroutine. This of course requires $\Theta(n^2)$ bits of memory, while in streaming we only have $\widetilde{O}(n)$.

Other results for ℓ_0 Best-Fit Ultrametrics. The other results for ℓ_0 Best-Fit Ultrametrics are pivot-based and do not work in our case. Indeed, one of them [CG24] is based on a large LP for which no streaming solution is known, while the other one [CFLDM22] is combinatorial but with approximation factor $\Omega(\log n)$.

1.3.2 Our techniques

ℓ_0 **Best-Fit Ultrametrics.** Our streaming algorithm is a divisive algorithm. In the divisive framework, each level of the tree is defined by a distinct distance from the input, which allows each level to be visualized as an instance of the correlation clustering problem. In this instance, two vertices are connected if their distance is at most the threshold associated with that level; otherwise, they are not connected. Following this, different layers of the ultrametric tree are built by repeatedly applying a division strategy in a top-down fashion. Here, we highlight the techniques we develop to design a semi-streaming algorithm that uses only a single pass and computes an $O(1)$ approximation for the ℓ_0 Best-Fit Ultrametrics problem.

Distances Summary. The first fundamental challenge is identifying which distances should be preserved in the constructed ultrametric, given that the input may contain $\Omega(n^2)$ distinct distances. A divisive algorithm may need to perform its division strategy on every level defined by such a distance (of course, sometimes it may decide not to divide anything); however, in the semi-streaming setting, we cannot even afford to store all these distances. Instead, we work with a compressed set of distances that effectively captures all important information. More formally, we focus on distances d for which there exists at least one vertex u such that the number of vertices with distance less than d from u is significantly smaller than the number of vertices with distance at most d from u. Using this notion, we demonstrate that it is sufficient to consider only a near-linear number of distances to achieve a good approximation.

Agreement Sketches. A key component in many (flat) clustering algorithms (including the first algorithm for Correlation Clustering [BBC02], which inspired many others, such as [CLM+21, AW22, CFLDM22, BCC+24]) involves comparing the set of neighbors of two vertices. While our division strategy also builds on such comparisons, both the hierarchical nature and streaming constraints of our setting present unique challenges. In Correlation Clustering each vertex has only a single set of neighbors, however, in our hierarchical setting, each layer of the tree is associated with a different distance threshold, producing different sets of neighbors for a vertex. In the worst case, there can be $O(n)$ such sets, and building a sketch for each can require up to quadratic space.

To address this, we build a new sketch for a node only when its set of neighbors changes significantly. The intuition here is that if the neighborhood of a node has not changed substantially, then a precomputed sketch for a nearby neighborhood will suffice. However, implementing this in a semi-streaming setting, where distances between pairs can arrive in any arbitrary order, is challenging. Since we cannot store the distances to all other nodes from a given node simultaneously, identifying significant changes in a node's neighborhood

---

[3]Informally, when we solve Agreement Correlation Clustering we obtain a clustering $\mathcal{C}$ with all its clusters being dense and the property that there exists a near-optimal clustering $\mathcal{C}'$ such that every cluster of $\mathcal{C}'$ is a subset of some cluster in $\mathcal{C}$.

becomes difficult. To manage this, we develop a new technique that combines random sampling with a pruning strategy, ensuring that the overall space required to store all the sketches is $\tilde{O}(n)$.

In this approach, we build each sketch by randomly sampling nodes. Assuming the neighborhood size has dropped substantially, we expect the correct sketch to reach a certain size. Notably, the set of neighbors of a node only shrinks as the distance decreases. Thus, for a specific weight threshold, if the sample (or sketch) size grows considerably, this indicates that the neighborhood has not changed much, so we disregard that weight threshold and delete the corresponding sample from the sketch. Specifically, for each node, we build and store sketches when the neighborhood size shrinks by a constant factor. Following this, we consider at most $\log n$ different sizes, and storing the sketch for all sizes takes only polylogarithmic space for a node. Therefore, the total space required to store the sketches for all the nodes is bounded by $\tilde{O}(n)$. Moreover, the sketches ensure that for each node u and each weight w, there exists a weight w' such that the neighboring nodes of u at w and w' differ very little, and we have built a sketch corresponding to the neighboring set at weight w'.

Across-Levels Correlations. In divisive algorithms, while building a new level of the ultrametric tree, the recursions performed depend on the divisions computed at previous recursion levels. In this sense, the divisive framework can be viewed as an adaptive adversary for the division strategy we need to perform. This is not an issue when deterministic division strategies are used (e.g. as in [CFLDM22]), but it becomes particularly problematic in our case, where we are forced to use random sketches because of the issue with the $\Omega(n^2)$ distinct distances.

The challenge here arises from the fact that for a given vertex we do not build a new sketch for each level of the tree. Instead, we only construct a sketch when the set of neighbors changes substantially. Consequently, multiple levels of the tree must reuse the same sketch, which increases the correlation among clusters at different levels. This makes it difficult to ensure concentration bounds when limiting the overall error.

To address this, our approach aims to "limit" the dependencies by ensuring our algorithm has only a logarithmic recursion depth (as opposed to the $\Omega(n^2)$ recursion depth in straightforward divisive approaches). This allows us to afford independent randomness by using a new sketch for each recursion depth. To reduce the recursion depth, we make the following observation: if the correlation clustering subroutine identifies a large cluster (e.g., containing a 0.99 fraction of the vertices), we can detect this cluster without explicitly applying the correlation clustering algorithm (thus omitting the requirement of using a sketch). This is because all vertices within this cluster have large degrees, while those outside have very small degrees. Therefore, it suffices to identify the vertices with small degrees and remove them to generate the new cluster. It is important to note that the degree calculation must consider the entire graph, not just the subgraph induced by the current cluster being considered. Otherwise, intra-recursion dependencies could be introduced, and thus the logarithmic recursion depth guarantee may not suffice.

Within-Level Correlations. Correlation issues do not only occur vertically (across levels), but also horizontally (within the same level), as most algorithms for correlation clustering compute a cluster, and then recurse on the rest of the elements. However in our case, such an adaptive construction may lead to the possibility of reusing sketches, making it difficult to ensure concentration.

We overcome these issues in several ways. First, we use these sketches to compute the agreement among vertices (i.e., computing the similarity between the set of neighbors of each pair of vertices) before we start constructing any clusters. Finally we propose an algorithm that is relying solely on our agreement sketches and is decomposed into independent events, thus only requiring us to consult the sketches of each layer only once. By using the agreements precomputation and our proposed clustering algorithm we ensure concentration while limiting the error for all clusters within a level.

S-**Structural Clustering.** Finally, as argued in Section 1.3.1, we need a division strategy that is different from the existing Agreement Correlation Clustering of [CFLDM22]. That is because it can be proven that solving Agreement Correlation Clustering on arbitrary subgraphs requires $\Omega(n^2)$ bits of memory.

Instead, we introduce S-Structural Clustering, which is inspired by Agreement Correlation Clustering. The key distinction is that now we require a clustering of S to satisfy the structural properties, while also considering edges with only one endpoint in S. This distinction is exactly what allows us to generalize our proposed algorithm to solve S-Structural Clustering by relying solely on the global neighborhoods of its vertices. Interestingly, the resulting time complexity of our general algorithm only depends on the size of the subgraph, as we compress all the necessary global information through our sketches. Finally, we remark

that both the construction of the sketches (Section 3.1) and the introduction of the S-Structural Clustering (Section 3.2.2) are two novel contributions of our work and could be of independent interest.

ℓ_0 **Best-Fit Tree-Metrics.** In [Kip23] it is shown how to reduce ℓ_0 Best-Fit Tree-Metrics to ℓ_0 Best-Fit Ultrametrics. In this approach, however, one needs to create n different instances of ℓ_0 Best-Fit Ultrametrics, which is not feasible in the semi-streaming model. In this work, we show that randomly solving a logarithmic number of these n different instances suffices.

Our initial approach requires 3 passes over the stream. One for a preprocessing step implicitly constructing the ℓ_0 Best-Fit Ultrametrics instances, one to solve these instances (and post-process them to extract trees that solve the original problem), and a final one to figure which one of the logarithmically many trees we need to output (the one with the smallest cost is picked).

We further improve the number of passes to 2, by eliminating the need for the final pass. To do that, we note that there are many trees with "tiny" cost related to the input; let A be the set containing these trees. By triangle inequality, all trees in A have "small" cost related to each other. If we create a graph with the trees as nodes, and an edge between two nodes when the cost relative to each other is small, we then show that with high probability this graph contains a big clique. Finally, we show that any node (corresponding to a tree) from a big clique is a good enough approximation to the original input, even if it is not in A.

ℓ_∞ **Results.** Regarding ℓ_∞ Best-Fit Ultrametrics, we show that the existing exact algorithm [FKW93] can be straightforwardly adapted to a 2-pass semi-streaming algorithm. Naturally, as the problem has been solved exactly, no research has focused on approximation algorithms. In this work we show that the solution to a related problem (ℓ_∞ Min-Decrement Ultrametrics) 2-approximates ℓ_∞ Best-Fit Ultrametrics. Then we adapt the exact solution for ℓ_∞ Min-Decrement Ultrametrics [FKW93] to obtain a single pass semi-streaming algorithm.

We also show that no single-pass semi-streaming algorithm can give a better-than-2 approximation, for otherwise we could compress any graph in $\widetilde{O}(n)$ space. Together, these results completely characterize ℓ_∞ Best-Fit Ultrametrics in the semi-streaming setting, regarding the optimal number of passes and the optimal approximation factor.

For ℓ_∞ Best-Fit Tree-Metrics, there exists a reduction to Ultrametrics [ABF+99], blowing up the approximation by a factor 3. Adapting it in the semi-streaming requires one additional pass through the stream. Using it with our 2-approximation for ℓ_∞ Best-Fit Ultrametrics (rather than with the exact algorithm, as done in [ABF+99]), we need 2 passes (instead of 3).

2 Preliminaries

We start by presenting useful notations we employ throughout the text. We use uv to denote an unordered pair $\{u, v\}$. We use the term distance matrix to refer to a function from $\binom{V}{2}$ to the non-negative reals. Let D be a distance matrix. For easiness of notation, we use $w_{max} = \max_{uv} D(uv)$. We slightly abuse notation and say that for any $u \in V$, $D(uu) = 0$. For $p \geqslant 1$, $\|D\|_p = \sqrt[p]{\sum_{uv \in \binom{V}{2}} |D(uv)|^p}$ is the ℓ_p norm of D. We extend the notation for $p = 0$. In this case, $\|D\|_0$ denotes the number of pairs uv such that $D(uv) \neq 0$. We even say $\|D\|_0$ is the ℓ_0 norm of D, despite ℓ_0 not being a norm.

If T is a tree and u, v are two nodes in T, then we write $T(uv)$ to denote the distance between u and v in T. An ultrametric is a metric (V, D) with the property that $D(uv) \leqslant \max\{D(uw), D(vw)\}$ for all $u, v, w \in V$. It holds that (V, D) is an ultrametric iff there exists a rooted tree T spanning V such that all elements of V are in the leaves of T, the depth of all leaves is the same, and $D(uv) = T(uv)$ for all $u, v \in V$. We call trees with these properties *ultrametric trees*.

In the semi-streaming model, the input is again a distance matrix D on a vertex set V. Let $n = |V|$. Our algorithm has $\widetilde{O}(n)$ available space, and the entries of D arrive one-by-one, in an arbitrary order, as pairs of the form $(uv, D(uv))$. For simplicity, we use the standard assumption that each distance $D(uv)$ fits in $O(\log n)$ bits of memory.

We let E_w be the set of pairs uv such that $D(uv) \leqslant w$. We define $N_w(u)$, the set of neighbors of u at level w, to be the vertices v such that $uv \in E_w$ (including u itself), and the degree of u at level w to be

$d_w(u) = |N_w(u)|$. We even write $N(u)$ and $d(u)$ (instead of $N_w(u)$ and $d_w(u)$) when E_w is clear from the context. Given an ultrametric tree T, a cluster at level w is a maximal set of leaves such that every pairwise distance in T is at most w. It is straightforward that a cluster at level w corresponds to the set of leaves descending from a node of T at height $w/2$. Abusing notation, and only when it is clear from the context, we refer to this node as a cluster at level w as well.

Regarding ℓ_0, it is sufficient to focus on ultrametrics where the distances between nodes are also entries in D. That is because if an ultrametric T does not have this property, we can create an ultrametric T' with this property such that $\|T' - D\|_0 \leqslant \|T - D\|_0$ (folklore). To do this, simply modify every distance d in T to the smallest entry in D that is at least as large as d (if no such entry in D exists, then we modify d to be the maximum entry in D).

3 ℓ_0 Ultrametrics

In this section, we show how to $O(1)$-approximate ℓ_0 Best-Fit Ultrametrics with a single pass in the semi-streaming model. Formally we show the following.

Theorem 1. *There exists a single pass polynomial time semi-streaming algorithm that w.h.p. $O(1)$-approximates the ℓ_0 Best-Fit Ultrametrics problem.*

Our algorithm consists of two main phases. In the streaming phase, we construct efficient sketches that capture the essential information of the input matrix D. That is, we store a compressed representation of D, denoted as $\widetilde{D}$, which, unlike D, has a reduced size of $\widetilde{O}(n)$ rather than $O(n^2)$ values (hereafter called *weights*). Yet, we will show in Section 3.1 that for every weight $w \in D$ and every $u \in V$, a weight $\tilde{w} \in \widetilde{D}$ is stored, such that $N_w(u)$ and $N_{\tilde{w}}(u)$ are roughly the same. This guarantee enables us to approximate both the size of a neighborhood and the size of the intersection for two different neighborhoods.

The second step is a post-stream process that carefully utilizes the precomputed sketches while addressing the adaptivity challenges discussed in Section 1.3.2. In Section 3.2 we show how to compute the S-Structural Clustering subroutine, which we will use as our division strategy. In Section 3.3 we present our main algorithm, which uses this subroutine and the distances summary as black-boxes to construct the ultrametric tree. Finally, in Section 3.4, we establish the necessity of approximation in the streaming setting by proving that computing an optimal solution requires $\Omega(n^2)$ bits of memory.

3.1 Construction of Sketches

This section outlines the process for constructing sketches that enable our algorithm's implementation. For now we consider large neighborhoods of size $\Omega(\log^4 n)$. While a similar approach was used in [CLM+21][4], for the problem of correlation clustering, the challenge here is different. Unlike correlation clustering, where each vertex has only a single set of neighbors, each layer of the tree in our context is associated with a different distance threshold. Thus each varying threshold can produce a different set of neighbors for a vertex. In the worst case, there can be n such sets, and building a sketch for each changing set of neighbors for each vertex can require up to quadratic space (or even cubic, if implemented naively).

We denote the weight of an edge $e = uv$ by $w(e) = D(uv)$. Each sketch is constructed for a specific vertex with a predetermined size chosen from the set $\mathbb{S} = \{n, \frac{n}{(1+\zeta)}, \frac{n}{(1+\zeta)^2}, \ldots, \log^4 n\}$, where ζ is a small constant parameter to be adjusted. Each sketch will encapsulate a neighborhood of the vertex of size roughly s, and allow us to compare the common intersection of two different neighborhoods. Let w_s^v be the largest weight for which $\frac{s}{1+\zeta} < |N_{w_s^v}(v)| \leqslant s$. We call size s *relevant* for vertex v if such w_s^v exists.

To obtain the sketches, for each $s' \in \mathbb{S}$, we start by generating a random subset $R_{s'} \subseteq [n]$ by sampling each vertex from V independently with probability $\log^2 n/s'$, prior to processing the stream. For each vertex v, each relevant size s, and each s' satisfying $\frac{1}{2}s \leqslant s' \leqslant s$, we define a sketch $\mathcal{S}_{s,s'}^v$. Every sketch consists of

---

[4]In [CLM+21], the authors claim polylogarithmic size sketches for each vertex. However, we are unable to verify this. Specifically, the random set is constructed by selecting each vertex with probability $\min\left\{\frac{a \log n}{\beta j}, 1\right\}$, where a is a constant. Since j is at most $O\left(\frac{\log n}{\beta}\right)$, the probability of selecting a vertex is at least $\min\{\Omega(a), 1\}$, which is a constant. Thus, each random set is of size $\Omega(n)$. Therefore, for a vertex v with $|N(v)| = \Omega(n)$, the sketch size will be of size $\Omega(n)$.

(i) an estimate of the parameter w_s^v, denoted by $\tilde{w}_s^v$, and (ii) an (almost) random sample of $v \times N_{\tilde{w}_s^v}(v)$ of size $O(\log^2 n)$, along with the weight of each sampled edge. To achieve this, we store a collection of edges $C_1^v, \ldots, C_\ell^v \subseteq v \times N_{\tilde{w}_s^v}(v)$, where all edges in C_i^v have the same weight w_i^v, which we also store alongside C_i^v, and let C_ℓ^v be the collection corresponding to the largest weight. Furthermore, we ensure that the overall size of all collections $\sum_{i \in [\ell]} |C_i^v|$ is $O(\log^3 n)$ bits.

The purpose of incorporating two size parameters, s and s' into the sketch is to enable comparisons of neighborhoods that have slightly different, but relatively close, sizes. Yet, for simplicity, the reader may assume $s = s'$ for the following construction and claims. We now describe the process of constructing a specific sketch $\mathcal{S}_{s,s'}^v$ given the input stream:

1. Initialize $counter = 0$, $w_m = 0$.

2. If e is not incident on v, continue to the next edge.

3. Else, if $w_m \neq 0$ and $w(e) \geq w_m$, continue to the next edge.

4. Else if $u \notin R_{s'}$, where $e = (u, v)$, continue to the next edge.

5. Otherwise proceed as follows:

    (a) If there is a collection of edges C_i^v with an associated weight of $w(e)$, add the edge e to C_i^v. Otherwise, create a new collection C_i^v containing the edge e alongside $w(e)$.

    (b) Increase $counter$ by 1.

    (c) If $counter > (1 + \frac{\zeta}{2})\frac{s}{s'}\log^2 n$, delete C_ℓ^v the collection with the largest weight associated with it, set $w_m = w_\ell^v$ and $counter = counter - |C_\ell^v|$.

After processing all the edges, output $\mathcal{S}_{s,s'}^v = \bigcup_{i \in [\ell]}(C_i^v \times \{w_i^v\})$, furthermore if $s = s'$ we let $\tilde{w}_s^v = \max_{i \in [\ell]} w_i^v$ and call it a *governing weight* of the sketches parametrized by v and s. Namely, $\tilde{w}_s^v$ is the weight associated with the neighborhood a sketch parametrized by v and s is encapsulating. The next claim shows that this sketches can be stored in semi-streaming settings.

Claim 11. *The sketches $\mathcal{S}_{s,s'}^v$, where $v \in V$ and $s \in \mathbb{S}$, can be constructed and stored in $O(n \log^4 n)$ bits.*

Proof. First, note that the total space required to store all $R_{s'}$, where $s' \in \mathbb{S}$, is $O(n \log^3 n)$ bits. Next, each sketch $\mathcal{S}_{s,s'}^v$ stores at most $O(\log^2 n)$ edges, thus requires $O(\log^3 n)$ bits. Since we build $O(\log n)$ different sketches for each vertex, the overall space required is $O(n \log^4 n)$. □

The next claim demonstrates that for every w_s^v there is a sketch with governing weight $\tilde{w}_s^v$ such that $\left|N_{\tilde{w}_s^v}(v)\right|$ is a good approximation to $\left|N_{w_s^v}(v)\right|$.

Claim 12. *With high probability, for each vertex v and each relevant size $s \in \mathbb{S}$, we have $(1 - \zeta)\left|N_{w_s^v}(v)\right| \leq \left|N_{\tilde{w}_s^v}(v)\right| \leq (1 + \zeta)^2 \left|N_{w_s^v}(v)\right|$.*

Proof. First, we consider the case where $\tilde{w}_s^v \leq w_s^v$, that is, $N_{\tilde{w}_s^v}(v) \subseteq N_{w_s^v}(v)$. We prove that with high probability, $|N_{w_s^v}(v) \backslash N_{\tilde{w}_s^v}(v)| \leq \zeta |N_{w_s^v}(v)|$.

Otherwise, if $|N_{w_s^v}(v) \backslash N_{\tilde{w}_s^v}(v)| > \zeta |N_{w_s^v}(v)|$, we claim that at least one of the following two bad events must occur. We define the first bad event as B_1, where no edge is sampled from $(v \times N_{w_s^v}(v)) \backslash (v \times N_{\tilde{w}_s^v}(v))$. We define the second bad event as B_2, where more than $(1 + \frac{\zeta}{2})\log^2 n$ edges are sampled from $v \times N_{w_s^v}(v)$. If neither of these bad events occurs, then at least one edge e is sampled from $(v \times N_{w_s^v}(v)) \backslash (v \times N_{\tilde{w}_s^v}(v))$, where $\tilde{w}_s^v < w(e) \leq w_s^v$, and the associated collection of $w(e)$ is not deleted. Consequently, e should survive, contradicting the claim that $\tilde{w}_s^v$ is the maximum weight of an edge that is sampled and not deleted. Since s is relevant and $\frac{s}{1+\zeta} \leq |N_{w_s^v}(v)| \leq s$, both events B_1 and B_2 occurs with probability at most $1/n^{10}$ using Chernoff bound.

Next, we consider the case where $\tilde{w}_s^v > w_s^v$, and thus $N_{w_s^v}(v) \subset N_{\tilde{w}_s^v}(v)$. We prove that with high probability, $|N_{\tilde{w}_s^v}(v)| \leq (1 + \zeta)^2 |N_{w_s^v}(v)|$.

Otherwise, if $\left|N_{\tilde{w}_s^v}(v)\right| > (1 + \zeta)^2 \left|N_{w_s^v}(v)\right|$, we claim that the following bad event B must occur. We define B as the event where at most $(1 + \frac{\zeta}{2})\log^2 n$ edges are sampled from $v \times N_{\tilde{w}_s^v}(v)$. If more than $(1 + \frac{\zeta}{2})\log^2 n$

9

edges are sampled from $v \times N_{\tilde{w}^v_s}(v)$, then $\tilde{w}^v_s$ cannot be obtained. According to Chernoff bound, since $\left| N_{\tilde{w}^v_s}(v) \right| > (1+\zeta)^2 \left| N_{w^v_s}(v) \right| > (1+\zeta)s$, B occurs with probability at most $1/n^{10}$. Therefore, the probability that $\left| N_{\tilde{w}^v_s}(v) \right| > (1+\zeta)^2 \left| N_{w^v_s}(v) \right|$ is at most $1/n^{10}$.

As there are n different choices for v, and $O(\log n)$ choices for s, the claim holds for all w^v_s w.h.p. $\qquad\square$

We now extend this result for every weight w, and show how to obtain a sketch that is a good approximation to $N_w(v)$.

Claim 13. *For each vertex v and each weight w with $|N_w(v)| \geqslant \log^4 n$, we can report a sketch associated with size s and governing weight $\tilde{w}^v_s$, such that with high probability, $\frac{\left| N_{\tilde{w}^v_s}(v) \right|}{1+5\zeta} \leqslant |N_w(v)| \leqslant \frac{\left| N_{\tilde{w}^v_s}(v) \right|}{1-\zeta}$*

Proof. Let $\tilde{w}^v_+$ (resp. $\tilde{w}^v_-$) be the immediate governing weights above (resp. below) w within all the sketches of v. We count the number of sampled edges in the sketch associated with the weight $\tilde{w}^v_+$ of weight greater than w. If there are less than $4\zeta \log^2 n$ such edges then we report this sketch, and otherwise we report the sketch associated with the weight $\tilde{w}^v_-$.

If there are less than $4\zeta \log^2 n$ edges with weight greater than w in the sketch, then using Chernoff bound w.h.p we have $\left| N_{\tilde{w}^v_s}(v) \right| \leqslant (1+5\zeta) |N^v_w(v)|$.

However, if $\left| N_{\tilde{w}^v_+}(v) \right| > (1+5\zeta) |N^v_w(v)|$, that is, $\left| N_{\tilde{w}^v_+}(v) \right| - |N^v_w(v)| > 5\zeta |N^v_w(v)|$, then using Chernoff bound we deduce that w.h.p we report the sketch associated with $\tilde{w}^v_-$. We now prove that this sketch is a good approximation. Let $s \in \mathbb{S}$ be such that $\frac{s}{1+\zeta} < |N^v_w(v)| \leqslant s$. By definition, s is relevant for v, and $\frac{|N_{w^v_s}(v)|}{1+\zeta} < |N^v_w(v)| \leqslant |N_{w^v_s}(v)|$. Following Claim 12, there exists a sketch $(\tilde{w}^v_s, \mathcal{S}^v_s)$, such that w.h.p. $(1-\zeta)\left| N_{w^v_s}(v) \right| \leqslant \left| N_{\tilde{w}^v_s}(v) \right| \leqslant (1+\zeta)^2 \left| N_{w^v_s}(v) \right|$. Thus, w.h.p. there exist a sketch $\tilde{w}^v_s$ such that, $\frac{\left| N_{\tilde{w}^v_s}(v) \right|}{(1+\zeta)^3} \leqslant |N_w(v)| \leqslant \frac{\left| N_{\tilde{w}^v_s}(v) \right|}{1-\zeta}$. Since $\frac{\left| N_{\tilde{w}^v_+}(v) \right|}{1+5\zeta} > |N^v_w(v)|$, it must hold that $|N_w(v)| \leqslant \frac{\left| N_{\tilde{w}^v_-}(v) \right|}{1-\zeta}$. Overall, the reported governing weight satisfies w.h.p, $\frac{\left| N_{\tilde{w}^v_s}(v) \right|}{1+5\zeta} \leqslant |N_w(v)| \leqslant \frac{\left| N_{\tilde{w}^v_s}(v) \right|}{1-\zeta}$. $\qquad\square$

We conclude this section by providing another data structure for storing the nearest $2\log^4 n$ neighbors for each vertex v, denoted by $N_{\texttt{close}}(v)$. This will allow us to compare neighborhoods of small size. The implementation of $N_{\texttt{close}}(v)$ is done using a priority queue with predefined and fixed size $2\log^4 n$. We add every edge incident to v to the priority queue $N_{\texttt{close}}(v)$ together with the associated edge. This leads to the following claim.

Claim 14. *For each vertex v, $N_{close}(v)$ can be stored in $O(\log^5 n)$ bits, and it contains the nearest $2\log^4 n$ neighbors of v.*

In this section, we have outlined the construction of sketches with a total space of $\tilde{O}(n)$, showing that for each vertex v and weight w, we can report a sketch associated with governing weight $\tilde{w}^v_s$ that with high probability is a random sample of a neighborhood $N_{\tilde{w}^v_s}(v)$ that is roughly of the same size as $N_w(v)$. In the next section, we will demonstrate how these sketches can be utilized to estimate the size of the symmetric difference in a way that supports the algorithm's requirements, justifying the need for maintaining several sketches for each choice of v and s.

3.2 Structural Clustering

In this section, we introduce an algorithm that requires a single pass over the input stream to solve S-Structural Clustering. This extends the notion of Agreement Correlation Clustering from [CFLDM22], to which we refer as V-Structural Clustering. Our semi-streaming algorithm hinges on the key idea that clusters should be formed from vertices that share almost similar neighborhoods. We emphasize that our algorithm is also applicable in the standard RAM (non-streaming) setting and runs in near-linear $\tilde{O}(|S|^2)$ time, for $S \subseteq V$, improving the $\Omega(|V|^3)$ time algorithm previously known for V-Structural Clustering.

We begin our presentation in Section 3.2.1 with an algorithm solving V-Structural Clustering, designed to be adapted in the semi-streaming model. Then, in Section 3.2.2 we show how our proposed algorithm could also be extended to compute S-Structural Clustering, which is our division strategy for constructing

ultrametrics in Section 3.3. Finally, in Section 3.2.3 we show how to actually implement these algorithms in the semi-streaming model, by utilizing our sketches outlined in Section 3.1.

3.2.1 Algorithm for V-Structural Clustering (Agreement Correlation Clustering)

We begin by solving the Structural Clustering problem for the entire vertex set V. Our graph is $(V, E = E_W)$ for some weight W. First we present the definitions of *agreement* and *heavy* vertices as in [CLM$^+$21]. The parameters β and ϵ that appear in the following definitions are sufficiently small constants. Furthermore we denote by $\triangle$ the symmetric difference between two sets, that is, $A\triangle B = A\backslash B \cup B\backslash A$.

Definition 3.1 (agreement). *Two vertices u, v are in β-agreement iff $|N(u)\triangle N(v)| < \beta \max\{d(u), d(v)\}$, which means that u, v share most of their neighbors. $A(u)$ is the set of vertices in β-agreement with u.*

Definition 3.2 (heavy). *We say that a vertex u is ϵ-heavy if $|N(u)\backslash A(u)| < \epsilon d(u)$, which means that most of its neighbors are in agreement with u. Denote by $H(u)$ the ϵ-heaviness indicator of vertex u.*

Computing the β-agreement set $A(u)$ of a vertex and its ϵ-heaviness indicator $H(u)$ is a crucial part of the algorithm. Normally, both can be computed exactly by applying the definitions, even using a deterministic algorithm. However, in the semi-streaming model, we can only approximate $A(u)$ and $H(u)$ with high probability. In Section 3.2.3 we show that, using the sketches outlined in Section 3.1, we can achieve a sufficient approximation that allows us to solve the Structural Clustering for V with high probability.

We allow the following relaxations. Let $A(v)$ be a set containing all vertices that are in 0.8 agreement with v and no vertices that are not in β agreement with v. Similarly let $A^3(v)$ be a set containing all vertices that are in 2.4β-agreement with v and no vertices that are not in 3β-agreement with v. And finally let $H(v)$ be a method that returns true if v is ϵ-heavy and false if v is not 1.2ϵ-heavy.

With these tools and definitions at our disposal, we can introduce Algorithm 1. It is important to note that this algorithm is executed, using the sketches alone, post stream process. Given the respective sketches, the time complexity of the algorithm is $\widetilde{O}(|V|^2)$.

Algorithm 1 V-Structural-Clustering

1: **for** $v \in V$ **do**
2:      **if** $H(v)$ **and** v is not already included in an existing cluster **then**
3:          Create a new cluster $A^3(v)$
4: Create singleton clusters for all remaining vertices.

We next show that Algorithm 1 is guaranteed to return a set of disjoint clusters that satisfy the required structural properties. We are referring to the special properties of V-Structural Clustering, which are expressed in terms of the definitions of *important* and *everywhere dense* groups as in [CFLDM22].

Definition 3.3 (important group). *Given a correlation clustering instance, we say that a group of vertices C is important if for any vertex $u \in C$, u is adjacent to at least $(1 - \epsilon)$ fraction of the vertices in C and has at most ϵ fraction of its neighbors outside of C.*

Definition 3.4 (everywhere dense). *Given a correlation clustering instance, we say that a group of vertices C is everywhere dense if for any vertex $u \in C$, u is adjacent to at least $\frac{2}{3}|C|$ vertices of C.*

Lemma 15 formally outlines the supplementary properties required for a correlation clustering algorithm to qualify as structural, demonstrated in the context of Algorithm 1.

Lemma 15 (structural properties). *Suppose $\beta = 5\epsilon(1 + \epsilon)$ for a small enough parameter $\epsilon \leqslant 1/95$. Let $\mathcal{C}$ be the set of clusters returned by Algorithm 1. Then, for any important group of vertices $C' \subseteq V$, there is a cluster $C \in \mathcal{C}$ such that $C' \subseteq C$, and C does not intersect any other important groups of vertices disjoint from C'. Moreover, every cluster $C \in \mathcal{C}$ is everywhere dense.*

In order to prove Lemma 15, we require the following claims. The next fact follows immediately from the definition of agreement and will be utilized in the subsequent proofs of the claims (cf. [CLM$^+$21]).

Fact 16. *If u, v are in $i\beta$-agreement, for some $1 \leqslant i < \frac{1}{\beta}$, then*

$$(1 - i\beta)d(u) \leqslant d(v) \leqslant \frac{d(v)}{1 - i\beta}$$

Claim 17. *Suppose $(1 - 3\beta - 1.2\epsilon)(1 - 3\beta) > \frac{1}{2}$. Assume u_1, u_2 are two vertices for which $H(u_1)$ and $H(u_2)$ both return true. If u_2 is not part of $A^3(u_1)$, then the sets $A^3(u_1)$ and $A^3(u_2)$ are disjoint.*

Proof. Let v be a vertex common in both $A^3(u_1)$ and $A^3(u_2)$. Since u_1 is 1.2ϵ-heavy and in 3β-agreement with v, we have that v has at least $(1 - 3\beta - 1.2\epsilon)d(u_1)$ neighbors that are in β-agreement with u_1. Similarly, v has $(1 - 3\beta - 1.2\epsilon)d(u_2)$ neighbors that are in β-agreement with u_2. Using Fact 16, both are non-less than $(1 - 3\beta - 1.2\epsilon)(1 - 3\beta)d(v)$ and by assumption this is greater than $\frac{1}{2}d(v)$. Consequently, there is a vertex w in β-agreement with both u_1 and u_2.

Now, by the triangle inequality we get that u_2 is contained in $A^3(u_1)$:

$$|N(u_1) \triangle N(u_2)| < |N(u_1) \triangle N(w)| + |N(w) \triangle N(u_2)|$$
$$< \beta \max\{d(u_1), d(w)\} + \beta \max\{d(w), d(u_2)\} \leqslant 2.4\beta \max\{d(u_1), d(u_2)\}$$

Where the last inequality follows from Fact 16 and that $\beta \leqslant \frac{1}{6}$. $\square$

Claim 18. *Suppose $1.2\epsilon \leqslant 1/3 - 6\beta$. Every cluster C returned by Algorithm 1 is everywhere dense.*

Proof. Consider the cluster $C = A^3(h)$ created from the 1.2ϵ-heavy vertex h. We know that every $u \in C$ is in 3β-agreement with h. Also by Definition 3.2 of heavy vertices, h has at most 1.2ϵ fraction of its neighbors outside C, hence:

$$|N(h) \cap N(u) \cap C| \geqslant |N(h) \cap N(u)| - |N(h) \backslash C| \geqslant (1 - 3\beta - 1.2\epsilon)d(h)$$

This implies:

$$d(u, C) \geqslant |N(h) \cap N(u) \cap C| \geqslant (1 - 3\beta - 1.2\epsilon)d(h) \tag{1}$$

Next, we show $d(h)$ is an upper bound for the size of the component $|C|$. To this end, consider the set of vertices $B = N(h) \cap C$ and the set of edges E between B and $C \backslash B$. Every vertex $u \in C \backslash B$ outside of B is adjacent to at least $|B \cap N(u)| \geqslant (1 - 3\beta - 1.2\epsilon)d(h)$ vertices inside of B and thus $|E| \geqslant |C \backslash B|(1 - 3\beta - 1.2\epsilon)d(h)$. Moreover, every vertex $u \in B$ inside of B is adjacent to at most $3\beta \max(d(h), d(u)) \leqslant 3\beta d(h)/(1 - 3\beta)$ vertices outside of B, as deduced from Fact 16. It follows that $|E| \leqslant |B| 3\beta d(h)/(1 - 3\beta)$. By combining both inequalities we get:

$$|C \backslash B| \leqslant \frac{3\beta}{(1 - 3\beta)(1 - 3\beta - 1.2\epsilon)}|B| < \frac{3\beta}{1 - 6\beta - 1.2\epsilon}d(h)$$

Now, by adding up $|C \backslash B|$ with $|B|$ we obtain an upper bound on $|C|$ in terms of $d(h)$.

$$|C| = |C \backslash B| + |B| < \frac{3\beta}{1 - 6\beta - 1.2\epsilon}d(h) + d(h) = \frac{1 - 3\beta - 1.2\epsilon}{1 - 6\beta - 1.2\epsilon}d(h)$$

Together with Equation 1, we achieve the desired result.

$$d(u, C) \geqslant (1 - 3\beta - 1.2\epsilon)d(h) > (1 - 6\beta - 1.2\epsilon)|C| \geqslant \frac{2}{3}|C|$$

$\square$

Claim 19. *Suppose $0.8\beta \geqslant 2\epsilon\frac{2 - \epsilon}{1 - \epsilon}$. Let the pair of vertices u, v belong to the same important group, then u, v are in 0.8β-agreement.*

Proof. Suppose u, v belong to the same important group C. Through the properties of important groups, we get that, (i) both u, v are adjacent to at least $(1-\epsilon)|C|$ vertices of C, and thus disagree on at most $2\epsilon|C|$ vertices inside of C. (ii) u, v are adjacent to at most $\epsilon d(u), \epsilon d(v)$ vertices not in C, respectively. In total they disagree on at most:

$$|N(u) \triangle N(v)| \leqslant 2\epsilon|C| + \epsilon(d(u) + d(v)) \leqslant \frac{2\epsilon}{1-\epsilon} \max\{d(u), d(v)\} + 2\epsilon \max\{d(u), d(v)\}$$

$$\leqslant 2\epsilon \frac{2-\epsilon}{1-\epsilon} \max\{d(u), d(v)\} \leqslant 0.8\beta \max\{d(u), d(v)\}$$

Where the second inequality follows from Definition 3.3, a vertex in an important group has a degree that is at least $(1-\epsilon)$ fraction of C, that is, for any $u \in C$, $d(u) \geqslant (1-\epsilon)|C|$. $\qquad\square$

Claim 20. *Suppose $2\epsilon \leqslant 1 - 3\beta$. Let u, v belong to two disjoint important groups, then u, v are not in 3β-agreement.*

Proof. Say that u, v belong to two disjoint important groups C_u, C_v, respectively. Then by Definition 3.3, u has at least $(1-\epsilon)$ fraction of his neighbors in C_u, whereas v has at most ϵ fraction of his neighbors in C_u, which means that u, v disagree on at least $(1-\epsilon)d(u) - \epsilon d(v)$ neighbors inside C_u. Similarly, u, v disagree on at least $(1-\epsilon)d(v) - \epsilon d(u)$ neighbors inside C_v. Overall, the difference in their neighborhoods is:

$$|N(u) \triangle N(v)| \geqslant (1 - 2\epsilon)(d(u) + d(v)) > (1 - 2\epsilon) \max\{d(u), d(v)\}$$

The claim now follows directly from the Definition 3.1 together with the assumption that $1 - 2\epsilon \geqslant 3\beta$. $\qquad\square$

We are finally ready to prove Lemma 15.

Proof. Following Claim 17 and Claim 18 the clusters returned by the algorithm are disjoint and everywhere dense. Next, we show the properties related to important groups. First, let u be a vertex in some important group C, then u has at least a $1 - \epsilon$ fraction of its neighbors within C, and according to Claim 19, it is in 0.8β-agreement with all vertices in C. Thus, every vertex that belongs to an important group is ϵ-heavy, which also implies that any such vertex is part of a non-singleton cluster.

Now, assume u, v belong to the same important group C, and that u belongs to a cluster $A^3(h)$ created in step 3 of the algorithm, we will show that $C \subseteq A^3(h)$. Because u, v are in 0.8β-agreement, u also belongs to the set $A^3(v)$. However, the intersection of $A^3(h)$ and $A^3(v)$ at vertex u implies, based on Claim 17, that vertex v necessarily belongs to the cluster $A^3(h)$.

It remains to prove that if u and v belong to disjoint important groups, they cannot be part of the same cluster. Suppose, contrarily, that both u and v belongs to $A^3(h)$. Note that u is ϵ-heavy as it belongs to some important group. As such, since u in $A^3(h)$, the sets $A^3(u)$ and $A^3(h)$ are not disjoint (as both contains u). According to Claim 17, as both u, h are ϵ-heavy, h must also be in $A^3(u)$. Similarly, h belongs to $A^3(v)$ as well. Thus, $A^3(u)$ and $A^3(v)$ intersect at h. However, by Claim 17, this intersection implies that u and v must be in 3β-agreement, contradicting Claim 20. $\qquad\square$

3.2.2 S-Structural Clustering

So far we have provided an algorithm for V-Structural Clustering. However, what we really need for ℓ_0 Best-Fit Ultrametrics is the more general problem of S-Structural Clustering as defined in Lemma 21. Note that it is different from Agreement Correlation Clustering on the induced subgraph $G[S]$, because S-Structural Clustering also considers edges with only one endpoint in S.

Lemma 21. *Suppose $\beta = 5\epsilon(1+\epsilon)$ for a small enough parameter $\epsilon \leqslant 1/95$. For every $S \subseteq V$ we can output a set of clusters $\mathcal{C}$. This clustering ensures that for every important group of vertices $C' \subseteq S$, there is a cluster $C \in \mathcal{C}$ such that $C' \subseteq C$, and C does not intersect any other important groups of vertices contained in $S \backslash C'$. Moreover, every cluster $C \in \mathcal{C}$ is everywhere dense.*

In the rest of this section we provide a construction of S-Structural Clustering by reducing the problem to V-Structural Clustering. We start by generalizing the definitions 3.1 and 3.2 presented earlier:

Definition 3.5 (subset agreement). *We say that two vertices $u, v \in S$ are in β-agreement inside S if $|N(u) \triangle N(v)| + 2|N(u) \cap N(v) \cap \overline{S}| < \beta \max\{d(u), d(v)\}$, which means that u, v share most of their neighbors inside S. Denote by $A_S(u)$ the set of vertices that are in β-agreement with u inside S.*

Definition 3.6 (subset heavy). *We say that a vertex $u \in S$ is ϵ-heavy inside S if $|N(u) \backslash A_S(u)| < \epsilon d(u)$, which means that most of its neighbors are in agreement with u inside S. Denote by $H_S(u)$ the ϵ-heaviness indicator of vertex u inside S.*

Note that definitions 3.5 and 3.6 are more general than definitions 3.1 and 3.2, since we can derive the latter ones by substituting $S = V$. The extra term $2|N(u) \cap N(v) \cap \overline{S}|$ has an intuitive meaning that will become clear later during the reduction. Similarly to the previous section we need to approximate the new sets $A_S(u), A_S^3(u)$ and $H_S(u)$, which we do in Section 3.2.3.

We finally prove Lemma 21 by reducing S-Structural Clustering, for a set S in the correlation clustering instance G, to V-Structural Clustering, in a specially constructed instance G_S. The instance G_S can be seen as a transformation of G that preserves all internal edges within S and replaces all external neighbors $v \in \overline{S}$ of internal vertices $u \in S$ with dummy vertices, ensuring that our subset-specific definitions of agreement 3.5 and heaviness 3.6 applied to G correspond exactly to the original definitions 3.1 and 3.2 applied to G_S. Therefore, to produce the desired S-Structural Clustering, it suffices to run Algorithm 1 on S using $A_S(u), A_S^3(u), H_S(u)$ as the parameters. The full proof is presented below.

Proof. Denote by $\mathcal{C}_S$ the clustering of S returned by Algorithm 1 when executed over the vertex set S using $A_S(u), A_S^3(u), H(u)$ as parameters. These parameters refer to the S-subset agreement sets 3.5 and S-subset heaviness indicator 3.6 calculated for each vertex $u \in S$ given our correlation clustering instance G. Also denote by G_S a new correlation clustering instance that contains all the vertices in S and some additional dummy vertices. Given the subset S, the instance G_S can be seen as a transformation of G with the following steps. First insert all the edges of G with internal endpoints $u, v \in S$ to G_S. Second, consider all the edges of G with an internal endpoint $u \in S$ and an external endpoint $v \in \overline{S}$. If u has any neighbor other than v in the instance G (that is $d(u) > 2$), then create a dummy vertex u_v and insert the edge with endpoints u, u_v to G_S.

Next we need to notice that $A_S(u)$, which is a β-agreement set for G inside S according to Definition 3.5, is also a β-agreement set for G_S according to Definition 3.1. By definition $A_S(u)$ is calculated for every internal vertex $u \in S$ of G_S. For the sake of clarity we also define $A_S(u) = \{u\}$ for every dummy vertex $u \in \overline{S}$ of G_S. Indeed consider any dummy vertex $u_v \in \overline{S}$ of G_S, which is only adjacent to its internal vertex $u \in S$, by construction of G_S. Since u has at least some neighbor other than u_v, then u_v is not in β-agreement with u by Definition 3.1. But u_v is not in β-agreement with any other vertex $w \neq u$ since u can be their only common neighbor ($N_S(u_v) = \{u_v, u\}$). It now suffices to show that any pair of (non singleton) internal vertices $u, v \in S$ of G_S is in β-agreement according to Definition 3.1 if and only if $u, v \in S$ of G are in β-agreement inside S according to Definition 3.5. But this is true since, by construction of G_S, the degrees $d(u), d(v)$ in G are equal to the degrees $d_S(u), d_S(v)$ in G_S and that $|N_S(u) \triangle N_S(v)| = |N(u) \triangle N(v)| + 2|N(u) \cap N(v) \cap \overline{S}|$.

Now we will prove that $\mathcal{C}_S$ along with the singleton vertices $u_v \in \overline{S}$ of instance G_S constitute a V-Structural Clustering for G_S, that is a clustering satisfying the structural properties of Lemma 15. Using the same argument as with $A_S(u)$, we see that $A_S^3(u)$ is a 3β-agreement set of $G_S(u)$ and using the definitions 3.2 and 3.6 we see that $H_S(u)$ is an ϵ-heaviness indicator of G_S. Just for the sake of clarity we also define $A_S^3(u) = \{u\}$ and $H_S(u) = $ false for every dummy vertex $u \in \overline{S}$ of G_S. Given that the parameters $A_S(u), A_S^3(u), H_S(u)$ are some proper agreement sets 3.1 and heaviness indicators 3.2 for the instance G_S, then running Algorithm 1 in the entire vertex set of G_S produces a V-Structural Clustering for G_S. But during the execution of the Algorithm 1, every dummy vertex $u_v \in \overline{S}$ of G_S will be eventually ignored. Indeed the algorithm will never iterate through u_v as $H(u_v) = $ false and the algorithm will never include u_v in a non trivial cluster as there is no set $u_v \in A_S^3(w)$. So it would be equivalent to first iterate through S to create clusters $\mathcal{C}_S$ and later iterate through $\overline{S}$ to create the remaining singleton clusters.

Finally we are ready to prove that $\mathcal{C}_S$ is the required S-Structural Clustering of G, that is a clustering of S satisfying the structural properties of Lemma 21. We start by observing that that a group of vertices $C' \subseteq S$ is important in G if and only if it is important in G_S. By construction of G_S any (non singleton) internal vertex $u \in S$ has the same total degree $d(u) = d_S(u)$ and set of internal neighbors $N(u) = N_S(u)$ in both graphs G, G_S. And since $C' \subseteq S$, every vertex $u \in C'$ is adjacent to the same fraction of the vertices in

14

$|C'|$ and the same fraction of its neighbors outside of C' in both graphs G, G_S, which proves the equivalence of Definition 3.3 in the two graphs. Now we know that every important group of vertices $C' \subseteq S$ of G is also important in G_S and subsequently from Lemma 15 there is a cluster $C \in \mathcal{C}_S$ such that $C' \subseteq C$. Also, cluster C does not intersect any other important groups of vertices of G contained in $S \backslash C'$, since otherwise it would intersect with the respective disjoint important groups of vertices of G_S, which would contradict Lemma 15. Lastly, any cluster $C \in \mathcal{C}_S$ is everywhere dense in both G and G_S, since $C \subseteq S$ has the exact same internal edges by construction of G_S. $\qquad\square$

3.2.3 Computing Agreements

Building on the algorithm from Section 3.2.2, we now describe its adaptation to the semi-streaming model. Since this algorithm only requires computing approximations to β-agreements in S and heaviness queries, we need to prove the following lemma:

Lemma 22. *The following statements hold with high probability:*

1. *For a given $\gamma \in \{\beta, 3\beta\}$ and every $u, v \in S$, we can output 'yes' if $|N(u)\triangle N(v)| + 2\left|N(u) \cap N(v) \cap \overline{S}\right| < 0.8\gamma \max\{d(u), d(v)\}$ and 'no' if $|N(u)\triangle N(v)| + \left|N(u) \cap N(v) \cap \overline{S}\right| \geqslant \gamma \max\{d(u), d(v)\}$.*

2. *For every $u \in S$, we can output 'yes' if $|N(u)\backslash A_S(u)| < \epsilon d(u)$ and 'no' if $|N(u)\backslash A_S(u)| > 1.2\epsilon d(u)$.*

Before proving the lemma we state the following corollary which is a direct consequence of Claim 13.

Corollary 23. *With high probability, for each vertex v and each weight w, there exists $\tilde{w}_s^v$, such that $\left|N_w(v)\triangle N_{\tilde{w}_s^v}(v)\right| \leqslant 5\zeta \left|N_w(v)\right|$.*

Proof. If $N_w(v) \subseteq N_{\tilde{w}_s^v}(v)$, then by Claim 13, $\left|N_w(v)\triangle N_{\tilde{w}_s^v}(v)\right| \leqslant \left|N_{\tilde{w}_s^v}\backslash N_w(v)\right| \leqslant 5\zeta \left|N_w(v)\right|$. Else, $\left|N_w(v)\triangle N_{\tilde{w}_s^v}(v)\right| \leqslant \left|N_w(v)\backslash N_{\tilde{w}_s^v}\right| \leqslant \zeta \left|N_w(v)\right|$. $\qquad\square$

Proof. We are now ready to prove Lemma 22. For the first item, let $d(v) \geqslant d(u)$ and consider the 3 possible cases: (i) $d(u), d(v) \leqslant 2\log^4 n$, (ii) $d(u) \leqslant \log^4 n$ and $d(v) > 2\log^4 n$, and, (iii) $d(u), d(v) \geqslant \log^4 n$.

In the first case case the entire neighborhoods of u and v are known and the query can be computed precisely. Whereas in the second case, $|N(u)\triangle N(v)| \geqslant d(v) - d(u) \geqslant \frac{d(v)}{2}$, this implies that, u, v cannot satisfy the conditions required for a 'yes' instance, and we report 'no'. Consequently, we remain with the third case which will require the sketching scheme outlined in Section 3.1.

Let s_u and s_v be the sizes reported by Claim 13 for u, and v respectively. Then, we have that w.h.p, $\frac{s_v}{1+5\zeta} \leqslant d(v) \leqslant \frac{s_v}{1-\zeta}$, and similarly for $d(u)$. Thus:

$$|N(u)\triangle N(v)| \geqslant d(v) - d(u) = d(v)\left(1 - \frac{d(u)}{d(v)}\right) \geqslant d(v)\left(1 - \frac{(1+5\zeta)s_u}{(1-\zeta)s_v}\right)$$

Consequently, if $1 - \frac{(1+5\zeta)s_u}{(1-\zeta)s_v} > 0.8\gamma$, then $|N(u)\triangle N(v)| > 0.8\gamma d(v)$, and u, v cannot satisfy the conditions required for a 'yes' instance in this lemma, and thus we report 'no'.

Else, it is the case that $\frac{s_u}{s_v} \geqslant (1-0.8\gamma)\frac{1-\zeta}{1+5\zeta} > \frac{1}{2}$, where the last inequality follows by selecting sufficiently small values for γ and ζ.

Note that, $|N(u)\triangle N(v)| = d(u) + d(v) - 2\left|N(u) \cap N(v)\right|$, hence:

$$|N(u)\triangle N(v)| + 2\left|N(u) \cap N(v) \cap \overline{S}\right| = d(u) + d(v) - 2\left|N(u) \cap N(v) \cap S\right| \tag{2}$$

Using s_u, s_v we can approximate $d(u), d(v)$, respectively, and obtain:

$$\begin{aligned} \frac{s_u}{1+5\zeta} + \frac{s_v}{1+5\zeta} &- 2\left|N(u) \cap N(v) \cap S\right| \\ &\leqslant |N(u)\triangle N(v)| + 2\left|N(u) \cap N(v) \cap \overline{S}\right| \\ &\leqslant \frac{s_u}{1-\zeta} + \frac{s_v}{1-\zeta} - 2\left|N(u) \cap N(v) \cap S\right| \end{aligned} \tag{3}$$

15

Thus, to estimate $|N(u)\triangle N(v)| + 2\left|N(u) \cap N(v) \cap \overline{S}\right|$, it is enough to estimate $|N(u) \cap N(v) \cap S|$. W.l.o.g. assume $s_u \geqslant s_v$. We will consider the sketches $\mathcal{S}^v_{s_v,s_v}$ and $\mathcal{S}^u_{s_u,s_v}$ (otherwise, consider the sketches $\mathcal{S}^v_{s_v,s_u}$ and $\mathcal{S}^u_{s_u,s_u}$). Using Corollary 23, $N(u)$ and $N_{\tilde{w}^u_{s_u}}(u)$ disagree on at most $5\zeta d(u)$ elements where $5\zeta d(u) \leqslant 5\zeta d(v) \leqslant 5\zeta\frac{s_v}{1-\zeta}$, and similarly for $N(v)$ and $N_{\tilde{w}^v_{s_v}}(v)$. Let $M = N_{\tilde{w}^v_{s_v}}(v) \cap N_{\tilde{w}^u_{s_u}}(u)$, then:

$$|N(u) \cap N(v) \cap S| - 10\zeta\frac{s_v}{1-\zeta} \leqslant |M \cap S| \leqslant |N(u) \cap N(v) \cap S| + 10\zeta\frac{s_v}{1-\zeta} \tag{4}$$

Define a random variable $X^S_{u,v} = \left|\mathcal{N}^v_{s_v} \cap \mathcal{N}^u_{s_v} \cap S\right|$. Recall that these sketches are constructed using the random set $R_{s_v} \subseteq V$, where each vertex of V is sampled independently at random with probability $\frac{\log^2 n}{s_v}$. By linearity of expectation we have:

$$\mathbb{E}[X^S_{u,v}] = \frac{\log^2 n}{s_v}|M \cap S|$$

We apply Chernoff bound to obtain w.h.p $(1-\zeta)\frac{s_v}{\log^2 n}X^S_{u,v} < |M \cap S| < (1+\zeta)\frac{s_v}{\log^2 n}X^S_{u,v}$.

Combining both Equation 3 and Equation 4 with the bounds on $|M \cap S|$, we get:

$$2\frac{s_v}{1+5\zeta} - \frac{20\zeta}{1-\zeta}s_v - 2(1+\zeta)\frac{s_v}{\log^2 n}X^S_{u,v}$$
$$\leqslant |N(u)\triangle N(v)| + 2\left|N(u) \cap N(v) \cap \overline{S}\right| \tag{5}$$
$$\leqslant (2+5\zeta)\frac{s_v}{1-\zeta} + \frac{20\zeta}{1-\zeta}s_v - 2(1-\zeta)\frac{s_v}{\log^2 n}X^S_{u,v}$$

Observe that by Equation 5, for some constant k, we can also write:

$$\left|(|N(u)\triangle N(v)| + 2\left|N(u) \cap N(v) \cap \overline{S}\right|) - (2s_v - 2\frac{s_v}{\log^2 n}X^S_{u,v})\right| \leqslant k\zeta s_v$$

The lemma now follows by selecting small enough parameter ζ relative to γ, namely, $\zeta < \frac{1}{10k}\gamma$. Based on this, we report 'yes' if $2s_v - 2\frac{s_v}{\log^2 n}X^S_{u,v} \leqslant 0.9\gamma s_v$, and 'no' otherwise.

For the second item, if $d(u) \leqslant 2\log^4 n$ the entire neighborhood is known and the query can be computed precisely. Else, let $\mathcal{N}^u_{s_u}$ be the vertices defining the edges incident on u in sketch $\mathcal{S}^u_{s_u,s_u}$ and define the random variable $Y_u = |\mathcal{N}^u_{s_u} \cap S \cap A_S(u)|$.

Observe that, $|N(u)\backslash A_S(u)| = |N(u)| - |N(u) \cap S \cap A_S(u)|$, and we can write:

$$\frac{s_u}{1+5\zeta} - |N(u) \cap S \cap A_S(u)| \leqslant |N(u)\backslash A_S(u)| \leqslant \frac{s_u}{1-\zeta} - |N(u) \cap S \cap A_S(u)| \tag{6}$$

Similarly to the first part of the lemma we estimate $|N(u) \cap S \cap A_S(u)|$. Using Corollary 23 we have:

$$\left|N_{\tilde{w}^u_{s_u}}(u) \cap S \cap A_S(u)\right| - 5\zeta\frac{s_u}{1-\zeta} \leqslant |N(u) \cap S \cap A_S(u)| \leqslant \left|N_{\tilde{w}^u_{s_u}}(u) \cap S \cap A_S(u)\right| + 5\zeta\frac{s_u}{1-\zeta} \tag{7}$$

Note that, $\mathcal{N}^u_{s_u}$ contains a random sample of $N_{\tilde{w}^u_{s_u}(u)}$, where each vertex is sampled with probability $\frac{\log^2 n}{s_u}$. Thus, by linearity of expectation, the expected value of Y_u is $\frac{\left|N_{\tilde{w}^u_{s_u}(u)} \cap S \cap A_S(u)\right|}{s_u}\log^2 n$. Using Chernoff bound, we obtain w.h.p that:

$$(1-\zeta)\frac{s_u}{\log^2 n}Y_u < \left|N_{\tilde{w}^u_{s_u}} \cap S \cap A_S(u)\right| < (1+\zeta)\frac{s_u}{\log^2 n}Y_u \tag{8}$$

We then report 'yes' if $s_u - \frac{s_u}{\log^2 n}Y_u \leqslant 1.1\epsilon s_u$, and 'no' otherwise. We conclude that:

$$\left||N(u)\backslash A_S(u)| - (s_u - \frac{s_u}{\log^2 n}Y_u)\right| \leqslant k\zeta s_u$$

For some constant k. The lemma now follows by selecting small enough parameter ζ relative to ϵ, namely, $\zeta < \frac{1}{10k}\epsilon$. □

We can now establish S-Structural Clustering in the semi-streaming model:

Theorem 24. *Suppose $\beta = 5\epsilon(1 + \epsilon)$ for a small enough parameter $\epsilon \leqslant 1/95$. Given access to the sketches of all vertices in S, and w.h.p., for every $S \subseteq V$ we can output a set of clusters $\mathcal{C}$. This clustering ensures that for every important group of vertices $C' \subseteq S$, there is a cluster $C \in \mathcal{C}$ such that $C' \subseteq C$, and C does not intersect any other important groups of vertices contained in $S \backslash C'$. Moreover, every cluster $C \in \mathcal{C}$ is everywhere dense.*

Proof. The algorithm outlined in Section 3.2.2 only requires the computation of polynomially many approximations to β-agreements in S and queries of heaviness in S; these can be computed with high probability using Lemma 22. Note that this only requires access to the sketches of vertices in S. The theorem now follows from Lemma 21. $\qquad\square$

3.3 Main Algorithm

To run our main algorithm it suffices to obtain access to certain black-boxes established in the previous sections. Ideally, we would like to have access to a summary of the input distances, to estimations of neighborhood sizes, and to be able to repeatedly compute S-Structural Clustering for instances given by an adaptive adversary. We show that even though we cannot generally guarantee the last requirement, it suffices to guarantee it for a particular (technical) type of adaptive adversary (see Lemma 36).

We first need the following definitions. For a weight $w \in D$, let $\widehat{w}$ be the smallest weight in $\widetilde{D}$ such that $\widehat{w} > w$. Similarly, for any w we let $\breve{w}$ be the largest value in $\widetilde{D}$ smaller than w. We say that $\widetilde{D}$ is a *compressed set* if for $w \notin \widetilde{D}$, $\widetilde{D}$ has the property that $d_w(u) \leqslant (1 + \delta)d_{\breve{w}}(u)$ for all u. Finally let $\widetilde{d_w(u)}$ be a function with $\widetilde{d_w(u)} \in [(1 - \lambda)d_w(u), (1 + \lambda)d_w(u)]$ for a sufficiently small constant λ.

Our algorithm (see Algorithm 2 for the pseudocode) is a divisive algorithm running S-Structural Clustering at each level to divide a cluster. However, it then performs a different division strategy for the largest cluster. This different strategy for the largest cluster allows us to guarantee that each vertex only participates in a logarithmic number of S-Structural Clustering computations, and is only possible if the size of the largest cluster has not dropped by a constant factor.

More formally, our algorithm takes as argument a set S (initially the whole V) and a distance w (initially the maximum distance). First, it creates a tree-node A at distance $w/2$ from the leaves, whose leaves-descendants are all the vertices in S. Then it uses an S-Structural Clustering subroutine, and for each cluster C' with size at most $0.99|S|$ it recurses on $(C', \breve{w})$. The roots of the trees created from each of these recursions then become children of A.

Subsequently, for the largest cluster C we perform the following postprocessing: Let $w' \leftarrow \breve{w}$ and $w'' \leftarrow \widetilde{w'}$.

- If there are at most $0.99|S|$ vertices u in S with large estimated degree $\widetilde{d_{w''}(u)}$ (larger than $0.66|S|$), then we recurse on (C, w'); the root of the tree created from this recursion becomes a child of A.

- Otherwise, we let R contain the vertices u whose estimated degree $\widetilde{d_{w''}(u)}$ is small (less than $0.65|S|$), and recurse on (R, w'). The root of the tree created from this recursion (let us call it A') becomes a child of A, and then we update $A \leftarrow A'$. Finally, we repeat the postprocessing again (but this time on $C \backslash R$ (instead of R) at level $\breve{w}$ (instead of w)).

The idea of the postprocessing is that nodes whose degree drops significantly cannot be in a huge cluster without a big cost. The challenging part is showing that keeping the rest of the nodes in C is sufficient.

In the rest of this section we provide the proof of our main result (Theorem 1) along with its required lemmas. We remind the reader that even though (for simplicity) Algorithm 2 explicitly stores the output distance between every pair of vertices, we cannot afford to do that in the semi-streaming model. That is why, in the proof of Theorem 1, we show how we can implicitly represent all these distances by storing a tree. From this point on, we let $T = \ell_0(V, w_{max})$ be the output of Algorithm 2.

We first provide two results: T is a valid ultrametric, and the depth of the recursion of Algorithm 2 is $O(\log n)$. Informally, the latter is crucial in order to limit the dependencies across different recursive calls, which in turn allows us to treat different recursive calls as independent from each other. Of course the components of the algorithm guaranteeing the $O(\log n)$ recursion depth also make the analysis of the algorithm different.

Algorithm 2 $\ell_0(S, w)$

1: Mark S at level w as a core cluster
2: **if** $|S| \leqslant 1$ **then return**
3: obtain $\mathcal{C} = \{C_1, \ldots, C_k\}$ using an S-Structural Clustering subroutine on $(V, E_{\breve{w}})$
4: **for all** u, v in different clusters of $\mathcal{C}$ **do** $T(uv) \leftarrow w$
5: **for all** $C' \in \mathcal{C}$ with $|C'| \leqslant 0.99|S|$ **do** $\ell_0(C', \breve{w})$
6: **if** $\exists C \in \mathcal{C}$ with $|C| > 0.99|S|$ **then**
7:      $w' \leftarrow \breve{w}, w'' \leftarrow \widetilde{w'}$
8:      **while** $|C| > 0.99|S|$ and $|\{u \in C \mid \widetilde{d_{w''}(u)} > 0.66|S|\}| > 0.99|S|$ **do**
9:          $R \leftarrow \{u \in C \mid \widetilde{d_{w''}(u)} < 0.65|S|\}$
10:          **for all** $u \in R, v \in C \backslash R$ **do**
11:              $T(uv) \leftarrow w'$
12:          $\ell_0(R, w')$
13:          $C \leftarrow C \backslash R, w' \leftarrow \widetilde{w'}, w'' \leftarrow \widetilde{w''}$
14:      $\ell_0(C, w')$

Lemma 25. $T = \ell_0(V, w_{max})$ *is a valid ultrametric.*

Proof. We inductively prove that for any three vertices u_1, u_2, u_3, the strong triangle inequality (characterizing ultrametrics) $T(u_1 u_2) \leqslant \max\{T(u_1 u_3), T(u_2 u_3)\}$ is satisfied. It trivially follows if $|S| = 1$.

Otherwise, for any three vertices u_1, u_2, u_3, if not all 3 of them are in the same cluster of $\mathcal{C}$, then by Line 4 at least two pairs have distance w in T. The other pair cannot get distance larger than w, thus the strong triangle inequality is satisfied.

If all 3 of them are in a cluster of $\mathcal{C}$ with size at most $0.99|S|$, then our claim holds inductively, when we recurse in Line 5.

If all 3 of them are in the unique cluster of $\mathcal{C}$ with size greater than $0.99|S|$, then either all 3 of them stay in C by the end of the while-loop (and thus inductively our claim holds when recursing in Line 14), or there is a first time when one of them (say u_1) is in R. Now:

- If at the same time all of them are in R, inductively our claim holds when we recurse in Line 12.

- Else, if one more (say u_2) is in R, then $T(u_1 u_3) = T(u_2 u_3) = w'$, and $T(u_1 u_2)$ can be at most w', therefore the strong triangle inequality is satisfied.

- Otherwise $T(u_1 u_2) = T(u_1 u_3) = w'$, and $T(u_2 u_3)$ can be at most w', therefore the strong triangle inequality is satisfied.

$\square$

To analyze the approximation factor of our algorithm, we first define a tree OPT' that is an $O(1)$ approximation of an optimal tree OPT, but has more structure. We then show that T is an $O(1)$ approximation of OPT', and therefore an $O(1)$ approximation of OPT as well.

Lemma 26. *In Algorithm 2, for any given $u \in V$ we have that the number of recursive calls $\ell_0(S, w)$ with $u \in S$ are $O(\log n)$.*

Proof. If we recurse in Line 5, or in Line 14 after having $|C| \leqslant 0.99|S|$, the size of the vertex-set S drops by a constant factor.

If we recurse in Line 14 while $|C| > 0.99|S|$, then it holds that $|\{u \in C \mid \widetilde{d_{w''}(u)} > 0.66|S|\}| \leqslant 0.99|S|$. When in the next recursion call we run S-Structural Clustering, we have that for any cluster C and any vertex u it holds $d_{\widetilde{w'}}(u) \geqslant \frac{2}{3}|C|$, or equivalently $|C| \leqslant 1.5 d_{\widetilde{w'}}(u)$. Now if C only contains vertices v with $\widetilde{d_{w''}(v)} > 0.66|S|$, we get $|C| \leqslant 0.99|S|$. Otherwise it contains a vertex u with $\widetilde{d_{w''}(u)} \leqslant 0.66|S|$, which implies $d_{\widetilde{w'}}(u) \leqslant 0.66|S|/(1 - \lambda)$, and thus $|C| \leqslant 0.99|S|/(1 - \lambda)$, which is less than $0.9999|S|$ for sufficiently

small λ. In all cases, the size of C drops by a constant factor, and therefore all subsequent recursive calls are called with a vertex-set which is a constant factor smaller than S.

If we recurse in Line 12, it holds that the size of R is at most $0.01|S|$, as R only contains vertices u with $\widetilde{d_{w''}(u)} < 0.65|S| < 0.66|S|$.

In all cases, after at most 2 recursive calls, the size of the vertex-set argument drops by a constant factor, and thus the claim follows. $\qquad\square$

Obtaining OPT' Let us now describe how to obtain OPT', given OPT. To make the exposition easier, we define some intermediary trees that are also constant factor approximations to OPT.

We first use the transformation from [CFLDM22] on OPT, to acquire OPT'_1. We write $OPT'_1 = f(OPT)$ to denote this transformation. It works as follows: We first set $OPT'_1 = OPT$, and proceed top-down. If a cluster C has way too many missing internal edges ($|\{uv|u, v \in C, D(uv) \geqslant w\}| > \frac{\epsilon^2|C|^2}{12.5}$), or way too many outgoing edges ($|\{uv|u \in C, v \notin C, D(uv) < w\}| > \frac{\epsilon^2|C|^2}{12.5}$) we set $OPT'(uv) = w$ for all $u, v \in C$. Viewing OPT'_1 as a tree, this corresponds to replacing the subtree rooted at C with a star, effectively separating all vertices in C into singletons in all lower levels. Then we again proceed top-down; as long as there exists a vertex u in a non-singleton cluster C at level w, with more than an ϵ fraction of its neighbors outside C, or less than $(1 - \epsilon)$ of its neighbors inside C, we set the distance of u to w. Viewing OPT'_1 as a tree, this corresponds to removing u from all lower level clusters, effectively making it a singleton in all lower levels.

From the construction of OPT' we get the following properties:

Lemma 27. *Given an ultrametric U, let $U' = f(U)$. It holds that:*

- $\|U' - D\|_0 = O(\|U - D\|_0)$.

- *every cluster in U' is either an important cluster in its respective level or a singleton.*

- *if $U'(u'v') = w$ for any u', v', then there exist u, v such that $U(uv) = w$.*

Proof. The first two claims follow from Lemma 3.3 of [CFLDM22], and the third claim follows directly by the construction of U'. $\qquad\square$

We then create OPT'_2, which only has distances that are in $\widetilde{D}$, by modifying OPT'_1. If $OPT'_1(uv) \notin \widetilde{D}$, we set $OPT'_2(uv) = \widehat{OPT'_1(uv)}$. Otherwise, we set $OPT'_2(uv) = OPT'_1(uv)$. We say we *destroy* a cluster in an ultrametric tree by connecting its children to its parent and then removing said cluster (note that destroying a cluster in an ultrametric tree preserves the ultrametric property). Viewing OPT'_2 as a tree, our transformation corresponds to destroying all clusters at levels that are not in $\widetilde{D}$, one by one. Finally, we apply the transformation of Lemma 27 again, to get $OPT' = f(OPT'_2)$. Given the tree view of OPT', it is straightforward to verify that it is indeed an ultrametric, as OPT is also an ultrametric.

We now establish structural properties of OPT'. Recall that a cluster $C \subseteq V$ is important by Definition 3.3 if each vertex $u \in C$ is adjacent to at least $(1 - \epsilon)$ fraction of vertices in C while having at most ϵ fraction of its neighbors outside C. This definition leads naturally to the following lemma:

Lemma 28. *Let C be an important cluster, and u be a vertex in C. Then $(1 - \epsilon)|C| \leqslant d(u) \leqslant |C|/(1 - \epsilon)$. Equivalently, for any $u \in C$ we have $(1 - \epsilon)d(u) \leqslant |C| \leqslant d(u)/(1 - \epsilon)$.*

Proof. Directly by the definition of an important cluster 3.3, u is connected with $(1 - \epsilon)|C|$ vertices in C, therefore $(1 - \epsilon)|C| \leqslant d(u)$. On the other hand, u can be connected with all vertices in C, and only an ϵ fraction of its edges can be out of C. Therefore $d(u) \leqslant |C| + \epsilon d(u)$, which means $d(u) \leqslant |C|/(1 - \epsilon)$. $\qquad\square$

We then prove the following properties:

Lemma 29. *For any uv we have that both $OPT'_2(uv), OPT'(uv) \in \widetilde{D}$.*

Proof. $OPT'_2(uv) \in \widetilde{D}$ follows directly by construction of OPT'_2. OPT' is obtained by modifying OPT'_2 without introducing any distances not in OPT'_2. $\qquad\square$

Lemma 30. *Every non-singleton cluster in OPT'_1, OPT' at level w is an important cluster of (V, E_w). Every non-singleton cluster in OPT'_2 at level w is a subset of some important cluster of (V, E_w).*

Proof. Every non-singleton cluster in OPT'_1 or in OPT' is an important cluster (not just a subset of one), directly by [CFLDM22] (Lemma 3.3, using ϵ instead of $\epsilon/8$).

As OPT'_2 is only splitting clusters of OPT'_1, the claim follows. $\qquad\square$

It holds that all described trees are $O(1)$ approximations of OPT.

Lemma 31. $\|OPT' - D\|_0 = O(\|OPT - D\|_0)$.

Proof. By Lemma 27, it suffices to show that $\|OPT'_2 - D\|_0 = O(\|OPT'_1 - D\|_0)$.

Let C at level w be a cluster of OPT'_1 that we modify in OPT'_2 (therefore $w \notin \widetilde{D}$, by construction).

We first prove that there are no two non-singleton clusters C_1, C_2 at level $\breve{w}$ in OPT'_1 such that $C_1 \subseteq C, C_2 \subseteq C$. Assume for the sake of contradiction that there exist such C_1, C_2, and w.l.o.g. $|C_1| \leqslant |C_2|$. By Lemma 30 we have that C_1, C_2, C are all important clusters in their respective levels. For any $u \in C_1$ we have $d_{\breve{w}}(u) \leqslant |C_1|/(1-\epsilon)$, by Lemma 28. As $u \in C$, again by Lemma 28 we have $d_w(u) \geqslant (1-\epsilon)|C| \geqslant 2(1-\epsilon)|C_1|$. But as $w \notin \widetilde{D}$, we have that $d_w(u) \leqslant (1+\delta)d_{\breve{w}}(u)$. But then it should be $d_{\breve{w}}(u) \geqslant 2(1-\epsilon)|C_1|/(1+\delta)$ and at the same time $d_{\breve{w}}(u) \leqslant |C_1|/(1-\epsilon)$, which is a contradiction for sufficiently small δ, ϵ.

Therefore, the only difference between OPT'_1, OPT'_2 is that certain nodes become singletons at some consecutive levels $w_1 > \ldots > w_k \notin \widetilde{D}$ of OPT'_2 (for which $\widetilde{w_1} = \ldots = \widetilde{w_k}$), while they were already singletons at level $\widetilde{w_1}$ in OPT'_1. Thus, for pairs including any such vertex u, the cost of OPT'_2 is increased by at most the number of outgoing edges of u in these levels, that is by $x = |\bigcup_{i=1}^k N_{w_i}(u)|$. But as $w_1 > \ldots > w_k$, we have $N_{w_1}(u) \supseteq \ldots \supseteq N_{w_k}(u)$, therefore $x = d_{w_1}(u) = O(d_{\widetilde{w_1}}(u))$ (due to $w_1 \notin \widetilde{D}$). However u was already a singleton at level $\widetilde{w_1}$ of OPT'_1, and thus OPT'_1 was paying $d_{\widetilde{w_1}}(u)$ for pairs including u. This proves our claim. $\qquad\square$

We now prove some structural properties of T related to OPT'. Informally:

- For every cluster C of OPT', there exists a cluster C_T of T at the same level.

- No cluster C_T of T contains two non-singleton clusters of OPT' of the same level.

- Every cluster of T is dense inside.

Lemma 32. *Let C be a cluster of OPT'. Then there exists a cluster $C' \supseteq C$ of T at the same level.*

Proof. Let OPT'_{sub} be a subtree of OPT', whose root corresponds to a cluster $A \subseteq V$ and is at level w. We prove an even stronger statement, namely that if we run Algorithm 2 with parameters (S, w) and obtain T_{sub}, where $S \supseteq A$, then for any cluster C of OPT'_{sub} there exists a cluster $C' \supseteq C$ of T_{sub} at the same level. This immediately implies the lemma, by setting $OPT'_{sub} = OPT'$ and $T_{sub} = T$ by running Algorithm 2 with parameters (V, w_{max}).

The claim immediately follows if $|C| = 1$. It also follows for the topmost cluster of OPT'_{sub}, as it corresponds to A while the root of T_{sub} at the same level corresponds to $S \supseteq A$.

Now assume C is a cluster of OPT'_{sub}, and let C_p be its parent cluster. Inductively, C_p is a subset of some cluster C'_p of T_{sub} at the same level.

If C'_p is a core cluster, then C is a subset of an important cluster by Lemma 30. As we obtain the children of C'_p by C'_p-Structural Clustering, we obtain a cluster C' containing the important cluster.

If C'_p is not a core cluster, we find a set $R \subseteq C'_p$, create cluster $C'_p \backslash R$ in T_{sub} at level w, and recurse on R at level $\widehat{w}$. Notice that, by Line 9, R only contains vertices u with $\widetilde{d_{w''}(u)} < 0.65|S|$ (where S is such that $0.99|S| < |C'_p| \leqslant |S|$), and there are at most $0.01|S|$ such vertices (Line 8). Therefore $0.98|S| < |C'_p \backslash R| \leqslant |S|$.

If $u \in R$, then $d_w(u) < \frac{1+\lambda}{1-\lambda}0.65|S| < 0.653|S|$ for sufficiently small λ. Similarly, at least $0.99|S|$ vertices in C'_p have degree larger than $0.657|S|$ at level w (Line 8).

Now if $|C| > 0.01|S|$, then it contains some vertex with degree larger than $0.657|S|$ at level w; as C is an important cluster (Lemma 30), all vertices inside it have degree above $0.653|S|$ (Lemma 28), and therefore completely lies in $C'_p \backslash R$. If $|C| \leqslant 0.01|S|$, then again by Lemma 28 it can only contain vertices with degree at most $0.02|S|$, therefore only contains vertices in R; as we recurse on R, we inductively prove our claim. $\qquad\square$

Lemma 33. *Let C be a non-singleton cluster of OPT' at level w. Then any $u \in C$ has $d_w(u) > 0.6|C|$.*

Proof. If C is obtained by S-Structural Clustering, it directly follows that u has $d_w(u) > 2|C|/3 > 0.6|C|$. Otherwise, C is obtained by removing all vertices with $\widetilde{d_w(u)} < 0.65|S|$ from its parent cluster C_p, for which we have $C_p \subseteq S$ for some vertex-set S. But then all vertices in C have $\widetilde{d_w(u)} \geqslant 0.65|S|$, which means $d_w(u) \geqslant \frac{1-\lambda}{1+\lambda}0.65|S| \geqslant \frac{1-\lambda}{1+\lambda}0.65|C|$, which implies our claim for sufficiently small λ. $\qquad\square$

Lemma 34. *Let C_1, C_2 be non-singleton clusters of OPT' at level w. There is no cluster C' of T at level w such that $C' \supseteq C_1 \cup C_2$.*

Proof. Assume for the sake of contradiction that this is not true, and that w.l.o.g. $|C_1| \leqslant |C_2|$. By Lemma 33 any vertex in C_1 has degree at least $0.6|C|$ at level w. But by Lemma 28 it has degree at most $|C_1|/(1-\epsilon) \leqslant 0.5|C|/(1-\epsilon)$, which is a contradiction for sufficiently small ϵ. $\qquad\square$

We are now ready to prove that T is a constant factor approximation of OPT.

Lemma 35. $\|T - D\|_0 = O(\|OPT - D\|_0)$.

Proof. By Lemma 31, it suffices to show that $\|T - D\|_0 = O(\|OPT' - D\|_0)$. Let E be the pairs uv for which $OPT'(uv) = D(uv) \neq T(uv)$. For all other pairs T pays at most as much as OPT'. In turn, it suffices to show $|E| = O(\|OPT' - D\|_0)$.

Let $uv \in E$. By Lemma 32, there exists a top level such that u, v are in the same cluster C in T but not in OPT'. By $OPT'(uv) = D(uv) \neq T(uv)$ we have that at this level u and v do not share an edge. By Lemma 34, one of the two nodes is a singleton in OPT'. Let $e(u)$ be the degree of u at the topmost level for which u is a singleton in OPT' but not in T ($e(u) = 0$ if no such level exists), and $c(u)$ be the cluster of T containing u in this level ($c(u) = \varnothing$ if no such level exists). By the above discussion, $|E| \leqslant \sum_{u \in V} |c(u)|$.

Notice that when u is a singleton in OPT' but not in T (as u is in a non-singleton cluster C' of T at that level), then u has degree at least $0.6|C'|$, by Lemma 33. Therefore $\|OPT' - D\|_0 = \Omega(\sum_{u \in V} |c(u)|)$, which proves our claim. $\qquad\square$

Lemma 36. *Assume that within a single pass in the semi-streaming model, we can:*

- *store a compressed set $\widetilde{D}$ of size $\widetilde{O}(n)$,*

- *store information of size $\widetilde{O}(n)$ that allows us to compute a $\widetilde{d_w(u)}$, for any vertex u and weight $w \in \widetilde{D}$.*

- *store information of size $\widetilde{O}(n)$ that allows us to compute S_i-Structural Clustering for k instances $\{(V, E_{w_1}), S_1\}, \ldots \{(V, E_{w_k}), S_k\}$. Instance $\{(V, E_{w_i}), S_i\}$ is only revealed after we compute S_j-Structural Clustering for every instance $\{(V, E_{w_1}), S_j\}$ with $j < i$ and may in fact depend on all these instances and the S_j-Structural Clusterings we output. Further, it holds that $w_i \in \widetilde{D}$ for all i, and each vertex $u \in V$ is contained in $O(\log n)$ of all S_i.*

Then we can $O(1)$-approximate ℓ_0 Best-Fit Ultrametrics in a single pass in the semi-streaming model.

Proof. We simply run Algorithm 2.

Instead of explicitly storing the distances between every pair of vertices, we build a tree that induces these distances, in a top-down fashion. At any given point, each leaf in the tree is associated with a subset of V, such that these subsets form a partition of V. Initially we have a single node (the root) at height w_{max}, associated with V. When we have $|S| = 1$, we simply create a leaf (corresponding to the unique node in S) at level 0.

In Line 4, we simply create $\mathcal{C}$ many children for the current node, each one corresponding to a different $C \in \mathcal{C}$, and recurse in all but the largest one, in case its size is larger than $0.99|S|$. We only run S-Structural Clustering when $w \in \widetilde{D}$, and by Lemma 26 each vertex u is only contained in $O(\log n)$ S_i-Structural Clustering computations; therefore by assumption of the lemma, we can perform these S_i-Structural Clustering computations.

Similarly, in the while loop we have an active node (initially it is the unique cluster C with $|C| > 0.99|S|$) at some level w. Then we decide a set R using the assumptions of our lemma, recurse on R to create more children of our active node, and also create one more child associated with $C \backslash R$ at level $\breve{w}$. Then we set the active node to be equal to the node corresponding to $C \backslash R$, and continue the execution.

It directly follows that the distances set in the algorithm are exactly the distances induced by our tree, and that the total space usage is $\widetilde{O}(n)$. $\qquad\square$

We now prove our main theorem.

Theorem 1. *There exists a single pass polynomial time semi-streaming algorithm that w.h.p. $O(1)$-approximates the ℓ_0 Best-Fit Ultrametrics problem.*

Proof. It suffices to guarantee that with high probability we can store the information required by Lemma 36. Our algorithm stores sketches for each vertex, as described in Section 3.1, in a single pass. In fact, it stores $c \log n$ independent instances of these sketches, for a sufficiently large c. This requires $\widetilde{O}(n)$ space.

To compute S_i-Structural Clustering for k instances $\{(V, E_{w_1}), S_1\}, \ldots \{(V, E_{w_k}), S_k\}$ such that $w_i \in \widetilde{D}$ and each vertex $u \in V$ is present in $O(\log n)$ of all S_i, we employ Theorem 24. In particular, using only the sketches of vertices in S, we can compute S-Structural Clustering with high probability. As we store $c \log n$ independent sketches for each vertex, we can use different sketches for each computation. Note that we cannot reuse our sketches, due to the dependencies across the instances and the clusterings we output.

We now show how to compute $\widetilde{d_w(u)}$ (which approximates $d_w(u)$), for any vertex $u \in V$ and weight $w \in D$ (this is stronger than $w \in \widetilde{D}$ required by Lemma 36). If $d_w(u) < 2 \log^4 n$, then we can exactly compute it, as we explicitly store the $2log^4 n$ nearest neighbors of u. Otherwise, by Claim 13, we can report a sketch associated with size s and weight $\tilde{w}_s^u$, such that with high probability, $\frac{d_{\tilde{w}_s^u}(u)}{1+5\zeta} \leqslant d_w(u) \leqslant \frac{d_{\tilde{w}_s^u}(u)}{1-\zeta}$. Therefore, for sufficiently small ζ, we have a sufficient approximation of $d_w(u)$.

Finally, we obtain $\widetilde{D}$ by using all the weights stored in memory; it follows its size is $\widetilde{O}(n)$. To show that $\widetilde{D}$ is a compressed set with high probability, assume for the sake of contradiction that there exists a $w \notin \widetilde{D}$ and a vertex u such that $d_w(u) > (1+\delta)d_{\tilde{w}}(u)$. But as we proved in the previous paragraph, we have a weight $w' \in \widetilde{D}$ such that $d_w(u) \leqslant \frac{d_{w'}(u)}{1-\zeta}$. For ζ sufficiently smaller than δ, this implies that $w' > w$. For these to hold simultaneously, it must be that in the obtained sketch we only have vertices v with $D(uv) \leqslant \check{w}$ or $D(uv) > w$.

It suffices to show that there exist $\nu d_{w'}(u)$ vertices $v \in N_{w'}(u)$ with $D(uv) \in (\check{w}, w]$, for a sufficiently small constant ν. This is because, if this is true, then with high probability we sample at least one such vertex. On one hand, we have that at most $5\zeta d_{w'}(u)$ vertices with distance above w. On the other hand he have at least $\delta d_{\tilde{w}}(u)$ vertices with distance above $\check{w}$. If $\delta \check{w} > 6\zeta d_{w'}(u)$, then we have at least $\zeta d_w w'(u)$ vertices with distance in $(\check{w}, w]$. Otherwise we have $\check{w} \leqslant 6\zeta d_{w'}(u)/\delta$, meaning there are at least $(1 - 5\zeta - 6\zeta d_{w'}(u)/\delta)$ vertices with distance in $(\check{w}, w]$, for sufficiently small ζ. $\qquad\square$

3.4 Lower bounds

Lower bounds for the problem of correlation clustering in data streams were thoroughly examined in [ACG+21]. In this section, we add additional natural results on top of this work.

The main lower bounds proved in [ACG+21] were to the problem of testing if a given graph can be partition to clusters with optimal cost of 0, under various edge weighting schemes. This observation leads directly to a similar computational limitation for algorithms that merely verify whether a matrix is ultrametric.

Theorem 37. *Any randomized k-pass streaming algorithm that tests whether an input matrix is an ultrametric with probability greater than $\frac{2}{3}$ requires $\Omega(\frac{n}{k})$ bits.*

Proof. Follows directly from Theorem 15 in [ACG+21]. $\qquad\square$

The subsequent theorems have implications for the problem of correlation clustering in streaming settings. We show that any algorithm addressing the correlation clustering problem, whether aiming to produce the optimal clustering or merely to report the optimal score, requires the use of $\Omega(n^2)$ bits. This requirement holds true even if the algorithm is permitted unbounded running time over the input. This results then naturally translate to the ultrametric construction framework.

Proposition 38. *Any randomized one-pass streaming algorithm that solves the correlation clustering problem with probability greater than $\frac{2}{3}$ requires $\Omega(n^2)$ bits.*

Proof. To prove the theorem we show a reduction to the index problem, where Alice is given a random string $x \in \{0,1\}^{\binom{n}{2}}$ and Bob is given a random index $(i,j) \in \binom{n}{2}$ and they need to output $x_{i,j}$ using a one-way protocol from Alice to Bob. This problem is known to require $\Omega(n^2)$ bits of communication even for randomized protocols [Abl96].

Consider a protocol for the index problem where Alice exploits a one-pass algorithm for the correlation clustering problem and stream the edges, where the positive edges are $\{(u,v) \mid x_{uv} = 1\}$. After the vector x has been processed by Alice, Alice sends the content of the memory of the streaming algorithm to Bob.

Bob stream two artificial cliques C_1, C_2 of size $2n$ each, together with the $+$ edges connecting i to C_1 and j to C_2, i.e. all edges $(i, c_1), (j, c_2)$ for $c_1 \in C_1$ and $c_2 \in C_2$. Furthermore, Bob stream $-$ edges between C_1, C_2 and the remaining vertices in the graph; that is, (c, k) for $c \in C_1 \cup C_2$ and $k \in [n] \backslash \{i, j\}$. Finally, Bob stream exactly $\frac{(2n+1)^2 - 1}{2} +$ edges between C_1 and C_2 and set the remaining edges to $-$ edges.

Now, $x_{i,j} = 1$ if and only if more than half of the edges between $C_1 \cup \{i\}$ and $C_2 \cup \{j\}$ are positive. It means an exact solution would have the cluster $C_1 \cup C_2 \cup \{i, j\}$. That is, only if i and j are end up in the same cluster in the correlation clustering solution. □

As we will see in the following theorem, the space constraints remains also in the setting where the algorithm simply opt to report the cost of the clustering.

Proposition 39. *Any randomized one-pass streaming algorithm that maintains the cost of an optimal correlation clustering solution with probability greater than $\frac{2}{3}$ requires $\Omega(n^2)$ bits.*

Proof. We again use a reduction from the index problem in a similar fashion, however we change the way Bob treats Alice's message.

Bob first duplicate Alice's message and stream different information to each message.

For the first message, Bob create 2 artificial cliques C_i, C_j of size n and connect them solely to i, j, respectively. Consequently, Bob obtains the cost of the optimal clustering. Due to the size of C_i, C_j, in any optimal clustering i, j are in their newly added cliques.

For the second message, Bob create a single artificial clique $C_{i,j}$ of size $2n$ and connect it to both i, j. Again, due to the size of $C_{i,j}$ any optimal clustering contains i, j in a cluster including $C_{i,j}$. The cost is changed from the first message depending on whether i is connected to j, namely, The cost will decrease by 1 if and only if i and j are connected. Hence, by subtracting the cost of the first message from the cost of the second message Bob gets an indicator stating if $x_{i,j} = 1$ w.h.p. □

We conclude these results in the following theorem:

Theorem 3. *Any randomized single pass streaming algorithm that with probability greater than $\frac{2}{3}$ either solves the correlation clustering problem or maintains the cost of an optimal correlation clustering solution requires $\Omega(n^2)$ bits.*

Fitting an ultrametric to a similarity matrix that contain just two specific values for under the ℓ_0 or ℓ_1 norms is exactly the correlation clustering problem (cf. [AC11]). It follows that the above bounds also holds for fitting ultrametric for both ℓ_0 and ℓ_1.

Corollary 4. *For $p \in \{0, 1\}$, any randomized single pass streaming algorithm that with probability greater than $\frac{2}{3}$ either solves ℓ_p Best-Fit Ultrametrics or just outputs the error of an optimal ultrametric solution requires $\Omega(n^2)$ bits.*

4 ℓ_∞ Ultrametrics

In this section we provide a complete characterization of ℓ_∞ Best-Fit Ultrametrics in the semi-streaming model. We show that in a single round, this problem cannot be approximated with an approximation factor strictly smaller than 2, while a factor 2-approximation algorithm in a single round does exist. Finally, we show that in two rounds we can obtain an exact solution.

The lower bound result is derived from a reduction to the index problem in communication complexity. For the algorithmic results, we employ a reduction to the ℓ_∞ Min-Decrement problem, where we are only allowed to decrement the entries in the input matrix.

4.1 ℓ_∞ Ultrametrics lower bound

Theorem 6. *Any randomized one-pass streaming algorithm for ℓ_∞ Best-Fit Ultrametrics with an approximation factor strictly less than 2 and a success probability greater than $\frac{2}{3}$ requires $\Omega(n^2)$ bits of space.*

Proof. To prove the theorem, we show a reduction to the index problem, where Alice is given a random string $x \in \{0,1\}^{\binom{n}{2}}$ and Bob is given a random index $(i,j) \in \binom{\{n\}}{2}$ and they need to output $x_{i,j}$ using a one-way protocol from Alice to Bob. This problem is known to require $\Omega(n^2)$ bits of communication even for randomized protocols [Abl96].

Assuming such an algorithm to ℓ_∞ Best-Fit Ultrametrics, Alice streams the matrix D where $D(a,b) = x_{a,b} + 1$ for every $(a,b) \in \binom{\{n\}}{2}$.

Bob, equipped with the index (i,j), streams $D(n+1,i) = D(n+1,j) = 0$ and $D(n+1,k) = 1.5$ for $k \in [n]\backslash\{i,j\}$, and obtains the output.

Now, in the case that $x_{i,j} = 0$, consider the matrix $\bar{O}$ with $\bar{O}(n+1,i) = \bar{O}(n+1,j) = \bar{O}(i,j) = 0.5$ and $\bar{O}(a,b) = 1.5$ for all other indices. It is easy to verify that this is an ultrametric and that $\|D - \bar{O}\|_\infty = 0.5$, so $OPT \leqslant 0.5$.

However, in the case that $x_{i,j} = 1$ we will show that $OPT = 1$. Consider the matrix $\bar{O}$ with $\bar{O}(n+1,i) = \bar{O}(n+1,j) = \bar{O}(i,j) = 1$ and $\bar{O}(a,b) = 1.5$ for all other indices. Similarly to the previous case, this is an ultrametric and $\|D - \bar{O}\|_\infty = 1$, so $OPT \leqslant 1$. Let O be some optimal solution, thus $D(a,b) - OPT \leqslant O(a,b) \leqslant D(a,b) + OPT$ for every $(a,b) \in \binom{\{n\}}{2}$. Combining this with the ultrametric property of O, we get:

$$2 - \text{OPT} = D(i,j) - \text{OPT} \leqslant O(i,j) \leqslant \max\{O(n+1,i), O(n+1,j)\}$$
$$\leqslant \max\{D(n+1,i) + \text{OPT}, D(n+1,j) + \text{OPT}\} \leqslant \text{OPT},$$

Consequently $OPT = 1$. It follows that any randomized one-pass algorithm for ℓ_∞ Best-Fit Ultrametrics that claims an approximation factor strictly less than 2 would be capable of distinguishing between the two cases and correctly retrieving $x_{i,j}$ with good probability. $\square$

4.2 ℓ_∞ Ultrametrics algorithms

To solve ℓ_∞ Best-Fit Ultrametrics we will apply a reduction to the ℓ_∞ Min-Decrement Ultrametrics problem. In this variant, it is only allowed to decrement the entries in the input matrix. We will show that an optimal solution to this variant is 2-approximation to the best fit.

Lemma 40. *An optimal solution to the ℓ_∞ Min-Decrement Ultrametrics problem is at most 2 approximation to ℓ_∞ Best-Fit Ultrametrics.*

Proof. Let O denote an optimal solution to ℓ_∞ Best-Fit Ultrametrics given the input matrix D, where the optimal fitting cost $\|O - D\|_\infty$ is denoted by c. Consequently, we have $D \geqslant O - c$, now set $\bar{O}(i,j) = \max\{0, O(i,j) - c\}$, it follows that $\bar{O} \leqslant D$.

Note that $\bar{O}$ is also an ultrametric, for every i,j,k:

$$\bar{O}(i,j) = \max\{0, O(i,j) - c\} \leqslant \max\{0, \max\{O(i,k), O(k,j)\} - c\}$$
$$= \max\{\max\{0, O(i,k) - c\}, \max\{0, O(k,j) - c\}\} = \max\{\bar{O}(i,k), \bar{O}(k,j)\}$$

Additionally, $\bar{O}$ is a 2-approximation of O:

$$\|D - \bar{O}\|_\infty = \|D - \max\{0, O - c\}\|_\infty = \|D + \min\{0, -O + c\}\|_\infty$$
$$= \|\min\{D, D - O + c\}\|_\infty = \max_{i,j}\{\min\{D(i,j), D(i,j) - O(i,j) + c\}\}$$
$$\leqslant \max_{i,j} D(i,j) - O(i,j) + c \leqslant 2c$$

$\square$

Next, we will show that the ℓ_∞ Min-Decrement Ultrametrics can be derived from the Minimum Spanning Tree (MST). Similar ideas were used in Theorem 3.3 in [ABF⁺99].

Given an input matrix D, let T be an MST of D. T naturally yields an ultrametric by defining the distance between any two vertices i and j as the weight of the heaviest edge on the unique path connecting them, denoted henceforth by $T(i,j)$. This construction inherently satisfies the ultrametric property, as the tree structure ensures that there is exactly one path between any pair of vertices, thereby maintaining the ultrametric property.

Moreover, if the edge (i,j) of weight $D(i,j)$ is in T, then clearly $T(i,j) = D(i,j)$. If not, the edge forms a cycle with the edges of T. Given that T is an MST, it follows that $T(i,j) \leqslant D(i,j)$. Therefore, T is indeed a minimum decrement ultrametric of D.

Furthermore, every minimum decrement ultrametric has values smaller or equal to the values of T. To see this let $\bar{O}$ be an optimal minimum decrement ultrametric. For any edge (i,j), if (i,j) belongs to T then $T(i,j) = D(i,j)$ and since $\bar{O} \leqslant D$ it follows that $\bar{O}(i,j) \leqslant T(i,j)$. Else, let P be the path from i to j in T. Due to the ultrametric property, $\bar{O}(i,j) \leqslant \max_{(k,l)\in P} \bar{O}(k,l) \leqslant \max_{(k,l)\in P} D(k,l) = \max_{(k,l)\in P} T(k,l) = T(i,j)$.

Therefore, the minimum spanning tree provides an optimal solution to the minimum decrement problem. As noted in [FKM⁺05], the minimum spanning tree can be constructed in a single pass with $O(\log n)$ time per edge under the semi-streaming model. We conclude this result in the following lemma:

Lemma 41. *An optimal solution to the ℓ_∞ Min-Decrement Ultrametrics fitting problem can be constructed in a single pass over the stream with $O(\log n)$ time per edge.*

As proved in Proposition 6, this construction achieves the best possible approximation within a single pass over the stream. The next theorem is now an immediate consequence of Lemma 40 and Lemma 41.

Theorem 5. *There exists a single pass polynomial time semi-streaming algorithm that 2-approximates the ℓ_∞ Best-Fit Ultrametrics problem.*

We proceed to demonstrate that a second pass over the stream, while necessary, is also sufficient to achieve the optimal solution to ℓ_∞ Best-Fit Ultrametrics.

The implementation goes by first applying Theorem 5 to produce an ultrametric T. Then, in a second pass over the stream, simply compute the error of T on the input D, denoted by $\bar{c}$, and return $T' = T + \frac{\bar{c}}{2}$. The error of T' is $\frac{\bar{c}}{2}$, as $0 \leqslant D - T \leqslant \bar{c}$ it follows that $-\frac{\bar{c}}{2} \leqslant D - T - \frac{\bar{c}}{2} \leqslant \frac{\bar{c}}{2}$.

According to Lemma 40, $\bar{c}$ is at most twice the optimal cost. That is, the ultrametric $T' = T + \frac{\bar{c}}{2}$ achieves the optimal cost precisely. It also follows that the MST obtained from Theorem 5 provides precisely a 2-approximation of the optimal ultrametric and at the same time has the topology of the optimal solution. This now fully concludes the ℓ_∞ Best-Fit Ultrametrics problem in the streaming settings.

Theorem 7. *There exists a two-pass polynomial time semi-streaming algorithm that computes an exact solution to the ℓ_∞ Best-Fit Ultrametrics problem.*

5 ℓ_0 and ℓ_∞ Tree Metrics

The problem of ℓ_p Best-Fit Tree-Metrics is typically addressed through reduction to an ℓ_p Best-Fit Ultrametrics instance, introducing a constant multiplicative approximation factor. This reduction, first introduced for the ℓ_∞ norm, generalizes to every ℓ_p with $p \geqslant 1$ [ABF⁺99]. More recently, Kipouridis showed how to adapt this reduction to the ℓ_0 case as well [Kip23].

In this section we show how this methodology can be extended to the semi-streaming model. We will show that with just an additional pass over the input stream it is possible to construct the best fit tree metric. In what follows we will focus on ℓ_0 and ℓ_∞, aligning with the algorithms proposed in this paper. However, this method can be generalized for any ℓ_p norm with $p \geqslant 1$.

The reduction strategy involves selecting a pivot element a, for which let C^a denote the centroid metric defined by $C^a(i,j) = 2\max_{k\in[n]} D(a,k) - (D(a,i) + D(a,j))$. Using the best fit ultrametric algorithm we then obtain an ultrametric U^a of $D + C^a$ and an a-restricted tree of D by setting $T^a = U^a - C^a$. Where A is denoted an a-restricted metric of B if, $A(a,k) = B(a,k)$ for every $k \in [n]$ (in this context A, B are symmetric matrices). We will see that T^a is a constant approximation to the best fit tree metric.

25

Note that if all the values $D(a) := (D(a,k))_{k \in [n]}$ are stored in memory, it is possible to adjust the input distance matrix D as the stream is processed and ultimately compute T^a in a single pass. Consequently, the first pass is utilized for storing $D(a)$ for some predefined a, and the second pass computes the a-restricted tree metric T^a as outlined above.

Using this idea we will show how to solve the problem of tree metric fitting for both ℓ_0 and ℓ_∞.

5.1 ℓ_∞ Best-Fit Tree Metrics

In the case of ℓ_∞, any arbitrary selection of a pivot a will provide with a constant approximation factor. Let $\tilde{U}^a$ denote the 2-approximation ultrametric of $D + C^a$ obtained by the algorithm outlined in Theorem 5. Recall that this is a minimum decrement ultrametric.

We show the following lemma, similar ideas were also utilized in [ABF$^+$99].

Lemma 42. *For every element a, $T^a = \tilde{U}^a - C^a$ is a 2-approximation a-restricted tree metric.*

Proof. Let $m_a = \max_{k \in [n]} D(a,k)$. To see that T^a is a-restricted we will show that for every i, $\tilde{U}^a(a,i) = 2m_a$, it is then easy to verify that for every $i \in [n]$, $T^a(a,i) = (\tilde{U}^a - C^a)(a,i) = D(a,i)$. First note that, $(D + C^a)(a,i) = 2m_a$. Then, since $\tilde{U}^a$ is an MST of $D + C^a$, we have $\tilde{U}^a(a,i) = 2m_a$.

To show that T^a is a tree metric we will use the following claim as in [ABF$^+$99].

Claim 43. *For every $a \in [n]$, T is a tree metric if and only if $T + C^a$ is an ultrametric.*

From Claim 43 it immediately follows that T^a is a tree metric. It is left to show that T^a is a 2-approximation to any optimal a-restricted tree metric of D, denoted T^{OPT}.

$$\begin{aligned}
\|T^{OPT} - D\|_\infty &= \|(T^{OPT} + C^a) - (D + C^a)\|_\infty \\
&\geqslant \|U^{OPT} - (D + C^a)\|_\infty \quad \text{(by Claim 43, for some optimal ultrametric U^{OPT})} \\
&\geqslant \frac{1}{2}\|\tilde{U}^a - (D + C^a)\|_\infty = \frac{1}{2}\|T^a - M\|_\infty
\end{aligned}$$

Overall, $\|T^a - M\|_\infty \leqslant 2\|T^{OPT} - D\|_\infty$. □

Using Lemma 3.4 in [ABF$^+$99], an optimal a-restricted tree metric is 3-approximation of the optimal tree metric. Thus, as a consequence of Lemma 42, for any selection of pivot a, the output is a 6-approximation tree metric to the optimal tree metric. We summarize this in the following theorem:

Theorem 10. *There exists a two-pass polynomial time semi-streaming algorithm that 6-approximates the ℓ_∞ Best-Fit Tree-Metrics problem.*

5.2 ℓ_0 Best-Fit Tree Metrics

While in the ℓ_∞ case every selection of a pivot would result in a 6 approximation, this does not hold for ℓ_0; yet, Kipouridis proved the existence of $a \in [n]$ which achieves a 3-approximation. Kipouridis then executed n reductions, that included best ultrametric fit, to obtain the desired approximation tree metric.

Lemma 44 (Theorem 3 in [Kip23])**.** *A factor $\rho \geqslant 1$ approximation for ℓ_0 Fitting Ultrametrics implies a factor 6ρ approximation for ℓ_0 Fitting Tree Metrics.*

Since we cannot store every $D(a)$ in memory we will have to suggest a different scheme. We will show in the following lemma that for a randomly selected $a \in [n]$, obtaining an optimal a-restricted tree is a constant approximation to the optimal best fit tree.

Lemma 45. *If a is randomly selected then with probability $\geqslant \frac{3}{4}$ the resulting tree metric is at most 12 approximation to the ℓ_0 Best-Fit Tree-Metrics.*

Proof. Let T denote an optimal best fitting tree metric to D under ℓ_0. We have that, $OPT = \|D - T\|_0 = \frac{1}{2} \sum_{i \in [n]} \|D(i) - T(i)\|_0$. Fix $i \in [n]$, we will transform T to an i-restricted tree, and denote this as $T^{/i}$, note that this is not necessarily an optimal i-restricted tree on D. The transformation works by moving every j either toward or away from i until each j is at distance $D(i,j)$. This transformation is also described in Lemma 3.4 in [ABF+99] and Theorem 3 in [Kip23].

We next show that $T^{/i}$ is a good approximation of T. Observe that by moving j, only distances from/to j may be modified, thus at most n errors may be introduced. Moreover, j is moved only if $D(i,j) \neq T(i,j)$, so the number of errors introduced by this transformation is at most $n\|D(i) - T(i)\|_0$.

Summing over all i we get that:

$$\sum_{i \in [n]} \|T^{/i} - D\|_0 \leqslant \sum_{i \in [n]} OPT + n\|D(i) - T(i)\|_0$$

$$\leqslant n \cdot OPT + n \sum_{i \in [n]} \|D(i) - T(i)\|_0 \leqslant n \cdot OPT + n \cdot 2OPT = 3n \cdot OPT$$

So by randomly selecting $a \in [n]$ (with uniform distribution), the expected cost of an optimal a-restricted tree metric is at most $3OPT$. Then, through Markov's inequality, the probability that $\|T^{/a} - D\|_0 > 12OPT$ is less than $\frac{1}{4}$. □

In order to further improve the algorithm and obtain a high probability success rate, we sample not one but $t = \ln n$ pivots, $P = \{a_1, ..., a_t\}$, and obtain t trees $\{T^{a_1}, ..., T^{a_t}\}$, each is at most 12 approximation to the optimal fit with probability at least $\frac{3}{4}$ following Lemma 45. Let $\overline{OPT}$ be the minimum value such that the graph $G_{\overline{OPT}}$ over the vertex set P, with edges (a_i, a_j) where $\|T^{a_i} - T^{a_j}\|_0 \leqslant 24\overline{OPT}$, has a clique of size at least $\frac{1}{2}n$. Finally, we arbitrarily select some pivot a_i in that clique and return T^{a_i}.

Note that by definition, if $a < b$ then $G_a \subseteq G_b$. Hence, we can find $\overline{OPT}$ by carrying a binary search in the range of possible values of OPT, i.e. $\overline{OPT} \in [0, n^2]$. Since t is logarithmic in n, this entire process can be carried in semi-streaming settings.

Claim 46. *With high probability, any pivot selected from the outlined clique in $G_{\overline{OPT}}$ corresponds to at most 36 approximation of T.*

Proof. Consider G_{OPT} and let P' denote the set of all vertices in P that correspond to a 12 approximation of T. We first show that $|P'| \geqslant \frac{t}{2}$ w.h.p.

For every $a_i, a_j \in P'$, $\|T^{a_i} - T^{a_j}\|_0 \leqslant \|T^{a_i} - T\|_0 + \|T^{a_j} - T\|_0 \leqslant 24OPT$. It follows that there is an edge between every two vertices in P'.

Write $X = \sum_{i=1}^{t} X_i$ where X_i is the indicator that $a_i \in P'$, and let $\mu = \mathbb{E}[X]$. Following Lemma 45, $\mu \geqslant \frac{3}{4}t$.

Using Chernoff bound it holds that:

$$\mathbb{P}[X \leqslant (1 - \delta)\mu] \leqslant \exp(\frac{-\delta^2 \mu}{2}) = (\frac{1}{t})^{3\delta^2/8}$$

Let $\delta = \frac{1}{3}$ and obtain that w.h.p there is a clique in G_{OPT} of size at least $\frac{1}{2}t$.

Recall that, if $a < b$ then $G_a \subseteq G_b$, that is, w.h.p, $\overline{OPT} \leqslant OPT$. The probability that one of the vertices of the clique is at most 12 approximation of T is at least $1 - (1 - \frac{3}{4})^{t/2} = 1 - (\frac{1}{2})^t$. By selecting any pivot a_i in that clique with a corresponding tree T^{a_i} we have that w.h.p we report an a_i-restricted tree that is at most 36 approximation to T. Since w.h.p there exist T^{a_j} that is a 12 approximation to T, also, $\|T^{a_i} - T^{a_j}\|_0 \leqslant 24\overline{OPT} \leqslant 24OPT$. It Follows that:

$$\|T^{a_i} - T\|_0 \leqslant \|T^{a_i} - T^{a_j}\|_0 + \|T^{a_j} - T\|_0 \leqslant 24OPT + 12OPT \leqslant 36OPT$$

□

Together with Lemma 44 we obtain the theorem:

Theorem 8. *There exists a two-pass polynomial time semi-streaming algorithm that w.h.p $O(1)$-approximates the ℓ_0 Best-Fit Tree-Metrics problem.*

References

[ABF+99]   Richa Agarwala, Vineet Bafna, Martin Farach, Mike Paterson, and Mikkel Thorup. On the approximability of numerical taxonomy (fitting distances by tree metrics). *SIAM J. Comput.*, 28(3):1073–1085, 1999. Announced at SODA 1996.

[Abl96]   Farid Ablayev. Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science*, 157(2):139–159, 1996.

[AC11]   Nir Ailon and Moses Charikar. Fitting tree metrics: Hierarchical clustering and phylogeny. *SIAM J. Comput.*, 40(5):1275–1291, 2011. Announced at FOCS 2005.

[ACG+21]   Kook Jin Ahn, Graham Cormode, Sudipto Guha, Andrew McGregor, and Anthony Wirth. Correlation clustering in data streams. *Algorithmica*, 83:1980–2017, 2021.

[ACL+22]   Sepehr Assadi, Vaggos Chatziafratis, Jakub Lacki, Vahab Mirrokni, and Chen Wang. Hierarchical clustering in graph streams: Single-pass algorithms and space lower bounds. In Po-Ling Loh and Maxim Raginsky, editors, *Conference on Learning Theory, 2-5 July 2022, London, UK*, volume 178 of *Proceedings of Machine Learning Research*, pages 4643–4702. PMLR, 2022.

[Ard05]   Federico Ardila. Subdominant matroid ultrametrics. *Annals of Combinatorics*, 8:379–389, 2005.

[AW22]   Sepehr Assadi and Chen Wang. Sublinear time and space algorithms for correlation clustering via sparse-dense decompositions. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 10:1–10:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[BBA75]   Ronald L Breiger, Scott A Boorman, and Phipps Arabie. An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *Journal of Mathematical Psychology*, 12(3):328–383, 1975. doi:10.1016/0022-2496(75)90028-0.

[BBC02]   Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, page 238. IEEE Computer Society, 2002.

[BCC+24]   Soheil Behnezhad, Moses Charikar, Vincent Cohen-Addad, Alma Ghafari, and Weiyun Ma. Fully dynamic correlation clustering: Breaking 3-approximation. *CoRR*, abs/2404.06797, 2024.

[BCMT22]   Soheil Behnezhad, Moses Charikar, Weiyun Ma, and Li-Yang Tan. Almost 3-approximate correlation clustering in constant rounds. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 720–731. IEEE, 2022.

[BCMT23]   Soheil Behnezhad, Moses Charikar, Weiyun Ma, and Li-Yang Tan. Single-pass streaming algorithms for correlation clustering. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 819–849. SIAM, 2023.

[BDH+19]   Soheil Behnezhad, Mahsa Derakhshan, MohammadTaghi Hajiaghayi, Cliff Stein, and Madhu Sudan. Fully dynamic maximal independent set with polylogarithmic update time. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 382–405. IEEE Computer Society, 2019.

[Ber20]   Daniel Irving Bernstein. L-infinity optimization to bergman fans of matroids with an application to phylogenetics. *SIAM Journal on Discrete Mathematics*, 34(1):701–720, 2020.

[BL17]   Daniel Irving Bernstein and Colby Long. L-infinity optimization to linear spaces and phylogenetic trees. *SIAM Journal on Discrete Mathematics*, 31(2):875–889, 2017.

[CCL+24]   Nairen Cao, Vincent Cohen-Addad, Euiwoong Lee, Shi Li, Alantha Newman, and Lukas Vogl. Understanding the cluster linear program for correlation clustering. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1605–1616. ACM, 2024.

[CDK14]   Flavio Chierichetti, Nilesh N. Dalvi, and Ravi Kumar. Correlation clustering in mapreduce. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 641–650. ACM, 2014.

[CDK+21]   Vincent Cohen-Addad, Debarati Das, Evangelos Kipouridis, Nikos Parotsidis, and Mikkel Thorup. Fitting distances by tree metrics minimizing the total error within a constant factor. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 468–479. IEEE, 2021.

[CDL21]   Vincent Cohen-Addad, Rémi De Joannis De Verclos, and Guillaume Lagarde. Improving ultrametrics embeddings through coresets. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2060–2068. PMLR, 2021.

[CF00]   Victor Chepoi and Bernard Fichet. ℓ_∞-approximation via subdominants. *Journal of mathematical psychology*, 44(4):600–616, 2000.

[CFLDM22]   Vincent Cohen-Addad, Chenglin Fan, Euiwoong Lee, and Arnaud De Mesmay. Fitting metrics and ultrametrics with minimum disagreements. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 301–311. IEEE, 2022.

[CG24]   Moses Charikar and Ruiquan Gao. Improved approximations for ultrametric violation distance. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 1704–1737. SIAM, 2024.

[CGW05]   Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005. Announced at FOCS 2003.

[CKL20]   Vincent Cohen-Addad, Karthik C. S., and Guillaume Lagarde. On efficient low distortion ultrametric embedding. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 2078–2088. PMLR, 2020.

[CKL+24]   Mélanie Cambus, Fabian Kuhn, Etna Lindy, Shreyas Pai, and Jara Uitto. A $(3 + \varepsilon)$-approximate correlation clustering algorithm in dynamic streams. In *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 2861–2880. SIAM, 2024.

[CLLN23]   Vincent Cohen-Addad, Euiwoong Lee, Shi Li, and Alantha Newman. Handling correlated rounding error via preclustering: A 1.73-approximation for correlation clustering. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1082–1104. IEEE, 2023.

[CLM+21]   Vincent Cohen-Addad, Silvio Lattanzi, Slobodan Mitrovi'c, Ashkan Norouzi-Fard, Nikos Parotsidis, and Jakub Tarnawski. Correlation clustering in constant many parallel rounds. In *International Conference on Machine Learning*, pages 2069–2078. PMLR, 2021.

[CLMP22]   Vincent Cohen-Addad, Silvio Lattanzi, Andreas Maggiori, and Nikos Parotsidis. Online and consistent correlation clustering. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 4157–4179. PMLR, 2022.

[CLN22]   Vincent Cohen-Addad, Euiwoong Lee, and Alantha Newman. Correlation clustering with sherali-adams. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 651–661. IEEE, 2022.

[CLP+24]   Vincent Cohen-Addad, David Rasmussen Lolck, Marcin Pilipczuk, Mikkel Thorup, Shuyi Yan, and Hanwen Zhang. Combinatorial correlation clustering. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1617–1628. ACM, 2024.

[CMSY15]   Shuchi Chawla, Konstantin Makarychev, Tselil Schramm, and Grigory Yaroslavtsev. Near optimal LP rounding algorithm for correlationclustering on complete and complete k-partite graphs. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 219–228. ACM, 2015.

[CSE67]   L. L. Cavalli-Sforza and A. W. F. Edwards. Phylogenetic analysis models and estimation procedures. *The American Journal of Human Genetics*, 19:233–257, 1967.

[Day87]   William H.E. Day. Computational complexity of inferring phylogenies from dissimilarity matrices. In *Bulletin of Mathematical Biology*, volume 49(4), page 461–467, 1987.

[D'h05]   Patrik D'haeseleer. How does gene expression clustering work? *Nature Biotechnology*, 23:1499–1501, 2005.

[DHH+05]   Andreas Dress, Barbara Holland, Katharina T Huber, Jack H Koolen, Vincent Moulton, and Jan Weyer-Menkhoff. δ additive and δ ultra-additive maps, gromov's trees, and the farris transform. *Discrete Applied Mathematics*, 146(1):51–73, 2005.

[DMM24]   Mina Dalirrooyfard, Konstantin Makarychev, and Slobodan Mitrovic. Pruned pivot: Correlation clustering algorithm for dynamic, parallel, and local computation models. *CoRR*, abs/2402.15668, 2024.

[DPS+13]   Geet Duggal, Rob Patro, Emre Sefer, Hao Wang, Darya Filippova, Samir Khuller, , and Carl Kingsford. Resolving spatial inconsistencies in chromosome conformation measurements. *Algorithms for Molecular Biology*, 8(1):1–10, 2013.

[FKM+05]   Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2-3):207–216, 2005.

[FKW93]   Martin Farach, Sampath Kannan, and Tandy Warnow. A robust model for finding optimal evolutionary trees. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, pages 137–145, 1993.

[FRB18]   Chenglin Fan, Benjamin Raichel, and Gregory Van Buskirk. Metric violation distance: Hardness and approximation. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 196–209. SIAM, 2018.

[GGR+20]   Anna C. Gilbert, Albert Gu, Christopher Ré, Atri Rudra, and Mary Wootters. Sparse recovery for orthogonal polynomial transforms. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 58:1–58:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[GJ17] Anna C. Gilbert and Lalit Jain. If it ain't broke, don't fix it: Sparse metric repair. In *55th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2017, Monticello, IL, USA, October 3-6, 2017*, pages 612–619. IEEE, 2017.

[HKM05] Boulos Harb, Sampath Kannan, and Andrew McGregor. Approximating the best-fit tree under l_p norms. In *APPROX-RANDOM*, pages 123–133, 2005.

[Kip23] Evangelos Kipouridis. Fitting tree metrics with minimum disagreements. In *31st Annual European Symposium on Algorithms (ESA 2023)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2023.

[KLNHM17] Patrick K. Kimes, Yufeng Liu, David Neil Hayes, and James Stephen Marron. Statistical significance for hierarchical clustering. *Biometrics*, 73(3):811–821, 2017. doi:10.1111/biom. 12647.

[Kři88] Mirko Křivánek. The complexity of ultrametric partitions on graphs. *Information processing letters*, 27(5):265–270, 1988.

[KSVK20] Bipul Kumar, Arun Sharma, Sanket Vatavwala, and Prashant Kumar. Digital mediation in business-to-business marketing: A bibliometric analysis. *Industrial Marketing Management*, 85:126–140, 2020.

[LC05] Sanghoon Lee and M.M. Crawford. Unsupervised multistage image classification using hierarchical clustering with a bayesian similarity measure. *IEEE Transactions on Image Processing*, 14(3):312–320, 2005.

[LL09] Sebastian Lühr and Mihai Lazarescu. Incremental clustering of dynamic data streams using connectivity based representative points. *Data & knowledge engineering*, 68(1):1–27, 2009.

[LLM14] Pei Lee, Laks VS Lakshmanan, and Evangelos E Milios. Incremental cluster evolution tracking from highly dynamic network data. In *2014 IEEE 30th International Conference on Data Engineering*, pages 3–14. IEEE, 2014.

[MC23] Konstantin Makarychev and Sayak Chakrabarty. Single-pass pivot algorithm for correlation clustering. keep it simple! In *Advances in Neural Information Processing Systems*, volume 36, pages 6412–6421, 2023.

[MWZ99] Bin Ma, Lusheng Wang, and Louxin Zhang. Fitting distances by tree metrics with increment error. *Journal of combinatorial optimization*, 3:213–225, 1999.

[RGP08] Pedro Pereira Rodrigues, Joao Gama, and Joao Pedroso. Hierarchical clustering of time-series data streams. *IEEE transactions on knowledge and data engineering*, 20(5):615–627, 2008.

[SS62] Peter H.A. Sneath and Robert R. Sokal. Numerical taxonomy. *Nature*, 193(4818):855–860, 1962.

[SS63] Peter H.A. Sneath and Robert R. Sokal. Numerical taxonomy. the principles and practice of numerical classification. *Freeman*, 1963.

[War92] Harold Todd Wareham. *On the computational complexity of inferring evolutionary trees*. PhD thesis, Memorial University of Newfoundland, 1992.