Abstract—Given $x \in (\mathbb{R}_{\geq 0})^{(\binom{n}{2})}$ recording pairwise distances, the Metric Violation Distance problem asks to compute the $\ell_0$ distance between $x$ and the metric cone; i.e., modify the minimum number of entries of $x$ to make it a metric. Due to its large number of applications in various data analysis and optimization tasks, this problem has been actively studied recently.

We present an $O(\log n)$-approximation algorithm for Metric Violation Distance, exponentially improving the previous best approximation ratio of $O(\min \{n^{1/3}, \sqrt{n} \})$ of Fan, Raichel, and Van Buskirk [SODA, 2018]. Furthermore, a major strength of our algorithm is its simplicity and running time. We also study the related problem of Ultrametric Violation Distance, where the goal is to compute the $\ell_0$ distance to the cone of ultrametrics, and achieve a constant factor approximation algorithm. The Ultrametric Violation Distance problem can be regarded as an extension of the problem of fitting ultrametrics studied by Ailon and Charikar [SIAM J. Computing, 2011] and by Cohen-Addad, Das, Kipouridis, Parotsidis, and Thorup [FOCS, 2021] from $\ell_1$ norm to $\ell_0$ norm. We show that this problem can be favorably interpreted as an instance of Correlation Clustering with an additional hierarchical structure, which we solve using a new $O(\log n)$-approximation algorithm for correlation clustering that has the structural property that it outputs a refinement of the optimum clusters. An algorithm satisfying such a property can be considered of independent interest. We also provide an $O(\log n \log \log n)$ approximation algorithm for weighted instances. Finally, we investigate the complementary version of these problems where one aims at choosing a maximum number of entries of $x$ forming an (ultra-)metric. In stark contrast with the minimization versions, we prove that these maximization versions are hard to approximate within any constant factor assuming the Unique Games Conjecture.

Index Terms—Metric Violation Distance, Ultrametric, Approximation

I. INTRODUCTION

Numerous tasks arising in data analysis and optimization deal with data given as distances between pairs of objects. The triangle inequality, which simply states that “the distance from $i$ to $j$ cannot be strictly greater than the distance from $i$ to $k$ plus the distance from $k$ to $i$,” is arguably the most natural constraint one can expect from the set of pairwise distances. Formally in this paper, $x \in (\mathbb{R}_{\geq 0})^{(\binom{n}{2})}$ is called a metric if for every distinct $i, j, k \in [n]$, $x(i,j) \leq x(i,k) + x(j,k)$.

The work of the first, second and fourth authors is partially supported by the French ANR project ANR-18-CE40-0004-01 (FOCAL). The work of the third author is partially supported by a gift from Google.

Note that we allow distances to be zero. The metric structure is desirable in most machine learning and data analysis tasks where the input models dissimilarity between objects, e.g.: nearest neighbors, metric learning, and clustering.

The metric structure also allows for the existence of efficient algorithms with provable theoretical guarantees. For example, many central optimization problems in clustering and network design including $k$-median and Traveling Salesman Problem have no polynomial time algorithms with finite approximation ratios without the assumption that the given distances form a metric, but they admit constant factor approximation algorithms with the assumption. Furthermore, the metric structure has also helped develop faster algorithms (e.g., [1] uses the triangle inequality to accelerate $k$-Means).

However, when the distance data $x \in (\mathbb{R}_{\geq 0})^{(\binom{n}{2})}$ is obtained by measurements, it may be the case that the data does not form a metric due to noise of the measurement, missing data, and other corruptions. In this scenario, it is natural to find the metric $y$ closest to the given data. Assuming that the uncorrupted data should form a metric, $y$ can be considered as the denoised version of $x$ on which one can perform various tasks using the metric structure. For example, the mPAM matrices [2], which represent a certain measure of dissimilarity in protein sequencing, tend not to be distance matrices. However, the query can be accelerated in biological databases once a metric-based data indexing scheme can be constructed.

The measure of closeness we study in this paper is the $\ell_0$ distance $\|x-y\|_0 = |\{(i,j) \in \binom{n}{2} : x(i,j) \neq y(i,j)\}|$. This choice seems natural in practice because often pairwise distances are obtained by different (human) classifiers, and while most classifiers will do a good job, if one classifier makes an error, it may make a large amount; if $x$ originally forms a metric but one entry $x(i,j)$ is increased to a very large number, the $\ell_0$ objective will return the original metric where convex objectives like $\ell_1$ or $\ell_2$ may likely change every distance.

Metric Violation Distance: Hence we study the following Metric Violation Distance problem: Given $x \in (\mathbb{R}_{\geq 0})^{(\binom{n}{2})}$, find $y \in M_n$ that minimizes $\|x-y\|_0$, where $M_n \subseteq (\mathbb{R}_{\geq 0})^{(\binom{n}{2})}$ is the set of all metrics on $n$ points. Brickell, Dhillon, Sra, and Tropp [3] first formulated the problem with $\ell_p$ objectives for
There exists a randomized algorithm runs in time \( O(\log n) \) approximation algorithm for \( M_{IOLA} \)-distance. This is an exponential improvement over the previous best ratio of \( O(n^{2/3}) \) which is also significantly faster.

**Theorem 1.1:** There exists a randomized \( O(\log n) \)-approximation algorithm for \( M_{ISTRANCE} \)-distance that runs in time \( O(n^3) \).

Besides the improved approximation ratio, the strength of our algorithm is its simplicity. Note that the \( O(n^3) \) complexity is actually not cubic since the input has size \( \Omega(n^2) \). Furthermore, since in particular the algorithm can be used to verify whether the input data is already a metric (deciding whether the objective is 0 or not), the \( O(n^3) \) complexity is tight by a result of Williams and Williams [8] (assuming a standard fine-grained complexity conjecture).

Our algorithm is a simple pivot-based algorithm (see Algorithm 1) which, at each iteration, chooses a random pivot \( i \in [n] \) and minimally changes the values of \( x(j,k) \) so as to ensure that all the triangle inequalities involving \( i \) are satisfied. Then it recursively solves the problem on the smaller instance \([n]\setminus\{i\} \). While this algorithm is very similar in spirit to the pivot-based algorithms first introduced by Ailon, Charikar and Newman [9] for various ranking and clustering problems, our main innovation lies in the analysis: compared to the aforementioned problems, here the value of some distances might be modified multiple times throughout the algorithm, and thus the analysis requires different techniques, leading to a slightly worse approximation ratio (logarithmic vs. constant). We also provide an example showing that our analysis is tight.

**Ultrametric Violation Distance:** It is also natural to ask the same question for special classes of distances. One special class we study in this paper is the class of ultrametrics; \( x \in \mathbb{R}_{\geq 0}^{n \choose 2} \) is called an ultrametric if every distinct \( i,j,k \in [n] \), \( x(i,j) \leq \max(x(i,k),x(k,j)) \). Equivalently, \( y \in \mathbb{R}_{\geq 0}^{n \choose 2} \) is an ultrametric if and only if there exists a tree such that (1) each edge has a distance, (2) the distance from the root to every leaf is the same, (3) the leaves are labeled by \([n]\), and \( y \) is given by the tree distances between the leaves. Let \( U_n \) be the set of all ultrametrics on \( n \) points, and \( U_{ISTRANCE} \)-distance be the problem, where given \( x \), the goal is to find \( y \in U_n \) such that \( y - x \) is minimized. Since ultrametrics are closely related to hierarchical clustering, they lie at the heart of a large number of unsupervised learning approaches, such as for example the classic linkage algorithms, see the survey of [10]. There has thus been several results recently on computing ultrametric embeddings of input points minimizing various objectives ([11]–[25]).

We are not aware of any previous work on this ULTRAMETRIC VIOlATION DISTANCE problem with a \( \ell_0 \) objective, except that its APX-hardness can be easily deduced from the APX-hardness of CORRELATION CLUSTERING [26]. Ailon and Charikar [27] studied the \( \ell_1 \) objective version of the problem and gave a \( \min(T+2, O(\log n \log \log n)) \)-approximation where \( T \) is the height of the tree, improving on the \( O(T \log n) \)-approximation of Harb, Kannan, and McGregor [28]. Recently Cohen-Addad, Das, Kipouridis, Parotsidis, and Thorup [29] improved the approximation ratio to \( O(1) \) hence giving the first constant factor for the unweighted version of the problem in its full generality. There is also a large body of work on minimizing the maximum distortion, namely for the \( \ell_\infty \) objective, for which a 3-approximation is known since the 90s [30] and recent progress has been made when the input lies in a high-dimensional Euclidean space [31], [32]. Sidiropoulos, Wang and Wang [33] studied the outlier deletion version of the problem where the goal is to find the smallest \( S \subseteq [n] \) such that \( x \) induced by \([n]\setminus S \) becomes an ultrametric (among others), and gave a 3-approximation algorithm.

Our pivot-based approach for \( M_{ISTRANCE} \)-distance is robust, and as described in the full version [34], a minor modification of Theorem 1.1 yields a simple \( O(\log n) \)-approximation algorithm for ULTRAMETRIC VIOlATION DISTANCE as well. Furthermore, for this ultrametric variant, we show that our analysis is tight by proving that the algorithm cannot give better than an \( O(\log n) \)-approximation not only for random choices of pivots, but for any possible choice of a sequence of pivots, see Theorem 1.2 in the full version [34].

Our next result shows how to overcome this logarithmic barrier using different techniques. By seeing the problem as a hierarchical variant of CORRELATION CLUSTERING, we are able to develop a new constant-factor approximation for CORRELATION CLUSTERING, and leverage it to obtain an \( O(1) \)-approximation algorithm for ULTRAMETRIC VIOlATION DISTANCE.

**Theorem 1.2:** There exists a polynomial time deterministic \( O(1) \)-approximation algorithm for ULTRAMETRIC VIOlATION DISTANCE.

**Violation Distance for Weighted Instances:** For both \( M_{ISTRANCE} \)-distance and ULTRAMETRIC VIOlATION DISTANCE, it is also natural to study the extension of the problems to \emph{weighted instances}: the input consists of distances \( x \in \mathbb{R}_{\geq 0}^{n \choose 2} \) and weights \( w \in \mathbb{R}_{\geq 0}^{n \choose 2} \), and the goal is to find a metric or ultrametric \( y \in \mathbb{R}_{\geq 0}^{n \choose 2} \) to minimize \( \sum_{(i,j)} w(i,j) \cdot \|x(i,j) \neq y(i,j) \| \).

These more general versions may be harder to approximate than the unweighted version; there exist polynomial-time approximation-preserving reductions from both MULTICUT and LENGTH-BOUNDED CUT to \( M_{ISTRANCE} \)-distance on weighted instances, see Fan, Gilbert, Raichel, Sonthalia, and Van Buskirk [6]. The best approximation ratios
for them are $O(\log n)$ [35] and $O(n^{2/3})$ [36] respectively, and both problems are hard to approximate within any constant factor assuming the Unique Games Conjecture [37], [38]. The standard LP relaxation for LENGTH-BOUNDED CUT has an integrality gap of $\Omega(n^{2/3})$, so approximating the METRIC VIOLATION DISTANCE problem on weighted instances with a subpolynomial approximation ratio is already a major challenge.

For ULTRAMETRIC VIOLATION DISTANCE, the reduction from MULTICUT still holds, implying that it seems unlikely to have an $O(1)$-approximation for weighted graphs. However, we adapt the algorithm of Ailon and Charikar [27] to design an $O(\log n \log \log n)$-approximation algorithm even for weighted instances.

Theorem 1.3: There exists a polynomial time $O(\log n \log \log n)$-approximation algorithm for ULTRAMETRIC VIOLATION DISTANCE on weighted instances.

Maximization version and hardness: Finally, we also investigate the complexity of the complementary (maximization) versions of METRIC VIOLATION DISTANCE and ULTRAMETRIC VIOLATION DISTANCE problems, which we call MAX-METRIC VIOLATION DISTANCE and MAX-ULTRAMETRIC VIOLATION DISTANCE. Here the problem consists of a set of vertices $[n]$ and a set of pairs $(i,j) \in \binom{[n]}{2}$ with distance $x(i,j)$ and weight $w(i,j)$, and the goal is to compute an (ultra)metric $\text{dist} : \binom{[n]}{2} \to \mathbb{R}$ so as to maximize the total weight of pairs $(i,j)$ with $x(i,j) = \text{dist}(i,j)$. This maximization version has been studied in the context of molecular biology [7], where heuristic approximations algorithms have been provided. We prove that these problems are Unique Games-hard to approximate within any constant factor.

Theorem 1.4: Assuming the Unique Games Conjecture, it is NP-hard to approximate MAX-METRIC VIOLATION DISTANCE and MAX-ULTRAMETRIC VIOLATION DISTANCE within any constant factor.

This hardness result also holds for the instances that are unweighted and complete (see Corollary 5.15 and 5.16 in the full version [34]), and therefore stands in contrast with the minimization version of ULTRAMETRIC VIOLATION DISTANCE, for which our Theorem 1.2 provides a constant-factor approximation. For these maximization problems, we are not aware of any non-trivial approximation algorithms. Since these problems can be reformulated as choosing a maximum number of edges that avoid specific unbalanced cycles (see the summary of techniques below), a similarly-looking problem is the one of finding a subgraph with a maximum number of edges that does not contain a $k$-cycle for some fixed $k$. That problem has been studied by Kortsarz, Langberg and Zuzto [39], and features similar huge gaps between the best known approximation algorithms and the lower bounds.

Challenges and New Techniques

Pivot-based algorithms for (Ultra)Metric Violation Distance: From the outset, it might look like METRIC VIOLATION DISTANCE can be phrased as a simple hitting set problem for a family of violated triples, namely, the triples of distances that do not satisfy the triangle inequality. This would suggest an easy constant-factor approximation, by simply approximating the corresponding hitting set instance. However, it might happen that even though one finds a family of edges that hits all the bad triples, there is no way of changing the value of the distances on this family of edges without creating new bad triples. Such a behavior is pictured on the left of Figure 1 where we can remove two edges to hit all the violated triples, yet there is no way to choose corrected distances for them without inducing new metric violations. The culprit here is easily found: there is a violated 4-tuple, i.e., a cycle (when viewing the instance as a complete graph) where one distance is longer than the sum of the other distances on the cycle. This thus makes the METRIC VIOLATION DISTANCE problem considerably harder than a hitting set problem on bounded size cycles.

It was shown in [5] that hitting all of the violated cycles is equivalent to the METRIC VIOLATION DISTANCE problem, but there are potentially exponentially many of those (i.e.: $2^{O(\sqrt{n})}$ bad cycles of length $\sqrt{n}$, etc.), and thus no natural approximation algorithm exists. In [5], the authors provide an $O(OPT^{1/3})$ approximation algorithm by focusing primarily on bad cycles of length at most 6, which in particular requires enumerating those and has a high complexity cost.

In contrast, our approach in Algorithm 1 is to nevertheless focus on violated triples by using a pivot-based algorithm: We choose uniformly at random a pivot, freeze the values of its incident distances, and repair the potential violated triples incident to the pivot by changing the value of the third distance in such a triple. Then we recurse, i.e., pivot on another vertex until the entire metric has been repaired. The entire pseudocode is displayed in Algorithm 1.

The crux of the argument is to define the appropriate reparation: as the example on the right of Figure 1 shows, simply removing the offending edges from our graph is not enough, as it might not repair violated $k$-tuples for $k > 3$. Our choice in Algorithm 1 is to minimally modify the distance of

---

Fig. 1. Left: The two colored edges hit all the violated triples, but one cannot assign them new distance values without creating new violated triangles. Right: If we first pivot at D and remove the corresponding offending edges BE and AC, there are no violated triples anymore but no way to assign new distance values to AC and BE without creating new violated triangles.
Algorithm 1: MVD-PIVOT\( (n, x) \)

1. **Input:** \( n \in \mathbb{N}, x \in \mathbb{R}_{\geq 0}^{\binom{n}{2}} \). **Goal:** modify \( x \) to a metric.
2. **if** \( n \leq 2 \) **then**
   3. **return**
   4. **end**
5. **Choose the pivot** \( i \in [n] \) uniformly at random;
6. **for** \( j \neq k \in [n] \setminus \{i\} \) **do**
7. **if** \( x(j, k) > x(i, j) + x(i, k) \) **then**
   8. \( x(j, k) = x(i, j) + x(i, k) \);
   9. **end**
10. **if** \( x(j, k) < |x(i, j) - x(i, k)| \) **then**
11. \( x(j, k) = |x(i, j) - x(i, k)| \);
12. **end**
13. **end**
14. MVD-PIVOT\( (n - 1, x|_{\binom{\left\{ i\right\}}{2}}) \); // \( x|_{\binom{\left\{ i\right\}}{2}} \) is \( x \)
   restricted to \( \{ (j, k) \in \binom{\left\{ i\right\}}{2} \} \)
15. **return**

the offending edge so as to satisfy the triangle inequality (thus making it an equality). As we will show in Section 2, this choice yields very desirable properties (Lemma 2.2 and 2.3). However, an offending edge may be modified multiple times during the course of our algorithm, which gives it a different flavor compared to existing pivot algorithms. We bound this number of changes by a logarithmic quantity in Theorem 1.1, explain how it connects to the approximation ratio, and we also provide an example showing that this is tight. An interesting phenomenon here is that Ailon and Charikar [27] in their work on fitting ultrametrics under the \( \ell_1 \) objective only obtained an \( O(T) \)-approximation algorithm using pivot-based approach, where \( T \) is the number of different input distances. Here, we manage to use the structure of the problem together with a refined pivot algorithm so as to obtain an \( O(\log n) \)-approximation algorithm.

An almost identical algorithm and analysis also provides a logarithmic approximation ratio for Ultrametric Violation Distance. Furthermore, in the ultrametric setting, a very symmetric construction based on hypercubes allows us to pinpoint precisely the limitation of pivot-based techniques: we build an instance of Ultrametric Violation Distance where any choice of a sequence of pivots yields a solution that differs from the optimal by a logarithmic factor. In order to overcome this gap, this leads us to designing a completely different approach to obtain a better approximation factor for Ultrametric Violation Distance.

Ultrametric Violation Distance via Correlation Clustering: ULTRAMETRIC VIOLATION DISTANCE offers additional structure compared to Metric Violation Distance, as an ultrametric can favorably be interpreted as a hierarchical clustering: for any \( m \), the sets of points at distance smaller than \( m \) form natural equivalence classes and thus can be thought of as a clustering of the input.

This allows us to define a natural correlation clustering instance for finding groups of points that are distance less than \( m \) and groups of points that are at distance at least \( m \): place a “+” edge for any edge that has value strictly less than \( m \) and a “−” edge for any edge that has value at least \( m \). This defines an instance of correlation clustering on complete unweighted graphs; indeed, the goal of correlation clustering is to partition the input into groups such that the total number of “−” edges whose extremities are in the same group plus the total number of + edges whose extremities are in different groups is minimized.

Thus, for a given distance \( m \), the optimum solution for the whole Ultrametric Violation Distance problem induces a natural partitioning into groups (points at distance at most \( m \)) that implies a solution to the correlation clustering instance defined at level \( m \). The cost of the two solutions are similar. On the one hand, edges of length at least \( m \) whose extremities are both in the same group will be set to a distance smaller than \( m \) and so induce a cost of 1, and since they are “−” edges in the correlation clustering instance, they also induce a cost of 1. On the other hand, edges of length less than \( m \) whose extremities are in different groups, are at distance at least \( m \) in the Ultrametric Violation Distance solution and thus incur a cost of 1, and are “+” edges across clusters in our correlation clustering instance and also incur a cost of 1.

Thus, the Ultrametric Violation Distance problem can be cast as a sequence of correlation clustering instances where the goal is to find nested correlation clustering solutions where one aims at minimizing the number of violated edges (an edge being violated if it is a “+” edge at some level and its extremities are in different clusters at that level, or if it is a “−” edge at some level and its extremities are in the same cluster).

To solve instances of the above problem resulting from Ultrametric Violation Distance instances, we use a new \( O(1) \)-approximation algorithm for correlation clustering. It is worth noting that one cannot simply compute an \( O(1) \)-approximation to correlation clustering for the largest distance and recurse in each cluster. This is because “−” edges at the same distance are also minus edges at smaller distances and one may risk charging a “−” edge that is paid by the optimum solution at each distance until the extremities are separated by the algorithm. To avoid the above, we describe a new algorithm for correlation clustering that satisfies the property that the clusters it outputs do not mix the “important” clusters of the optimum solution and so essentially produce clusterings that are refinement of an (approximately) optimal solution. We can thus compute the ultrametric top-down distance by distance.

We also provide an \( O(\log \log n) \)-approximation algorithm for Ultrametric Violation Distance on weighted instances. It relies on the rounding of a natural linear programming relaxation for the underlying hierarchical clustering problem. The algorithm and its Seymour-style [40] analysis are heavily inspired by a similar algorithm of Ailon and Charikar [27] to solve the \( \ell_1 \) variant of the problem.
Hardness of the maximization versions: The previous algorithms (except the very last one) all rely extensively on the fact the input graph is complete. In contrast, when one studies the maximization versions of the Metric Violation Distance and Ultrametric Violation Distance problems, the complete case is as hard as the general case. Indeed, starting from an arbitrary instance, one can add edges of very high distance to make the graph complete while roughly preserving the value of the optimal solution. Furthermore, the Max-Metric Violation Distance problem can be encoded into a Max-Ultrametric Violation Distance problem by blowing up the distances. Our strategy to prove Theorem 1.4 is therefore to prove hardness of the Max-Ultrametric Violation Distance on general graphs, which is our main result. Here, the idea is to interpret the problem as a constraint satisfaction problem: an ultrametric distance can be interpreted as the distance between the leaves of a tree, and thus the Max-Ultrametric Violation Distance problem can be thought of as assigning one such leaf to each vertex so that the distances induced by the tree match the input distances as well as possible. In order to do so, we employ standard techniques for proving Unique Games-hardness of Max-CSPs (based on dictatorship tests and invariance principles, see for instance [41], [42]) but the soundness of the dictatorship test requires a specific analysis that strongly relies on the tree structure of the alphabet.

Roadmap: Section II is dedicated to the proof of our $O(\log n)$-approximation algorithm for our pivot-based algorithms for Metric Violation Distance and Ultrametric Violation Distance. Section III describes our $O(1)$-approximation for Ultrametric Violation Distance on unweighted complete inputs. Due to space restrictions, our pivot algorithm for Ultrametric Violation Distance, the matching lower bound, the $O(\log n \log \log n)$-approximation algorithm for weighted instances of Ultrametric Violation Distance, and the hardness results for the maximization versions are skipped and included to the full version [34].

II. An $O(\log n)$-Approximation for Metric Violation Distance and Ultrametric Violation Distance

In this section, we prove that Algorithm 1 provides an $O(\log n)$ approximation for the Metric Violation Distance problem. We then show how to modify it to also obtain an $O(\log n)$ approximation for the Ultrametric Violation Distance problem, and provide an example where any choice of pivot leads to a logarithmic approximation ratio.

We first introduce the following intuitive terminology. We view the input distance data $x$ as a family of weights on the edges of a complete graph. A triangle is a triple of vertices, or equivalently of edges. Let $ij$ be an edge belonging to a triangle $t = ijk$. If $x(ij) > x(ik) + x(jk)$, we say that the edge $ij$ is in excess. If $x(ij) < |x(ik) - x(jk)|$, we say that it is in deficit. In both cases we call the triangle $t$ unbalanced. After pivoting at a vertex $i$, the weights of the edges adjacent to $i$ never change in the remaining recursive calls, thus we say that they are frozen.

The intuition for the $O(\log n)$ approximation factor is the following. At each recursive call, the algorithm chooses uniformly at random a pivot vertex $i$ and repairs the metric around it in the following way: for any $j$, the distances $x(ij)$ are kept as is, and for each triangle $ijk$, the distance $x(ijk)$ is modified in the minimal way so that the triangle $ijk$ is no longer unbalanced. We first prove (Lemma 2.2) that such repaired triangles never become unbalanced again further down in the algorithm. Then the key difference with previous pivot-based algorithms [9], [27] is that throughout recursive calls, a same edge may get modified multiple times. The main idea of the algorithm is that due to the random choice of pivots, the set of distances that an edge can get assigned roughly shrinks by a half whenever it is modified, and thus (in expectation), a given edge gets modified $O(\log n)$ times. We use this to obtain a lower bound on the fractional packing for unbalanced triangles, which by linear programming duality lower bounds the optimal value of the solution.

We start with the following easy observation which directly follows from the description of Algorithm 1.

Observation 2.1: At the end of a recursive call of Algorithm 1 where $i$ was chosen as a pivot, none of the triangles adjacent to $i$ are unbalanced.

We denote by $T$ the set of all triangles, and by $T'$ the set of all unbalanced triangles. We first prove that Algorithm 1 makes progress, i.e., the set of unbalanced triangles shrinks as the algorithm progresses. The proof is a tedious but straightforward case analysis.

Lemma 2.2: At each execution of a pivoting step in Algorithm 1, no new unbalanced triangle is created.

In particular, combined with Observation 2.1, Lemma 2.2 shows that once a pivot has been made at a vertex $i$, its adjacent triangles are repaired and never become unbalanced again.

Proof: Let $i, j, k, m$ be a 4-tuple of vertices, and say that we pivot at $i$ during the algorithm. Pivoting at $i$ might change the weights of the edges $jk, km$ and $jm$. We denote by $x$ and $x'$ the weights respectively before and after pivoting. By construction, the triangles containing $i$ are not unbalanced after the pivot. If $jkm$ is unbalanced after the pivot, if it was already unbalanced, there is nothing to prove. Otherwise, the value of at least one of its edges was modified and is now in excess or in deficit for this triangle. If one of the vertices of $jkm$ had already been chosen as a pivot, say $m$, we argue inductively, i.e., we assume that none of the triangles adjacent to $m$ are unbalanced and prove that it is still the case with the new weights. The initialization of the induction is provided by Observation 2.1.

If the weight of an edge got increased and is in deficit, then necessarily another edge got increased and is in excess for this triangle, we look at that one instead. So if $jk$ is an edge that got increased and is in excess, without loss of generality, its new value is $x'(jk) = x(ij) - x(ik)$. Note that since the weight of $jk$ has changed, neither $j$ nor $k$ had been chosen before
as a pivot, as it would have frozen that edge. If neither \( jm \) nor \( km \) are frozen, the new weights of the edges \( jm \) and \( km \) satisfy \( x'(jm) \geq x(ij) - x(im) \) and \( x'(km) \geq x(im) - x(ik) \). If both were frozen, i.e., there had been a pivot at \( m \), then the weights of the edges \( jm \) and \( km \) satisfy \( x'(jm) = x(jm) \geq x(ij) - x(im) \) and \( x'(km) = x(km) \geq x(im) - x(ik) \) because the triangles \( ijm \) and \( ikm \) are not unbalanced by the induction hypothesis. Therefore \( x'(jk) \leq x'(jm) + x'(km) \) and \( jk \) is actually not in excess.

Likewise, if the weight of some edge got reduced and is in excess for this triangle, some other edge got reduced and is in deficit. So if \( jk \) is an edge that got reduced and is in deficit, its new weight is \( x'(jk) = x(ij) + x(ik) \). Again, neither \( j \) not \( k \) had been chosen before as a pivot. If neither \( jm \) nor \( km \) were frozen, the new weights of the edges \( jm \) and \( km \) satisfy \( x'(jm) \leq x(ij) + x(im) \) and \( x'(km) \geq x(im) - x(ik) \), and thus \( x'(jk) \geq x'(jm) - x'(km) \). If both were frozen, i.e., there had been a pivot at \( m \), then the weights of the edges \( jm \) and \( km \) satisfy \( x'(jm) \leq x(ij) + x(im) \) and \( x'(km) \geq x(im) - x(ik) \), because by the induction hypothesis, \( ijm \) and \( ikm \) are not unbalanced. Thus \( x'(jk) \geq x'(jm) - x'(km) \). Similarly, the other inequality \( x'(jk) \geq x'(km) - x'(jm) \) is satisfied, and thus \( jk \) is actually not in deficit.

The following lemma shows that when pivoting at a vertex \( i \), we are not only repairing the triangles adjacent to \( i \), but also other triangles which required smaller changes than the ones induced by \( i \).

**Lemma 2.3:**

1. Let \( e = uv \) denote an edge adjacent to \( k \) unbalanced triangles for which \( e \) is in excess. Order the third vertices of these triangles from \( 1 \) to \( k \) so that \( 1 \) is the vertex inducing the biggest decrease of the weight of \( e \) if it is chosen as a pivot. Then after pivoting at \( i \), all the triangles \( uvj \) for \( j \geq i \) are no longer unbalanced.

2. Let \( e = uv \) denote an edge adjacent to \( k \) unbalanced triangles for which \( e \) is deficit. Order the third vertices of these triangles from \( 1 \) to \( k \) so that \( 1 \) is the vertex inducing the biggest increase of the weight of \( e \) if it is chosen as a pivot. Then after pivoting at \( i \), all the triangles \( uvj \) for \( j \geq i \) are no longer unbalanced.

This lemma is less trivial than it might appear as the weights of \( uj \) and \( vj \) may have also changed with the pivoting at \( i \).

**Proof:** Throughout the proof, we denote by \( x \) the edge-weights before pivoting, and by \( x' \) the edge-weights after pivoting. Note that for both items, none of the edges \( uv \), \( uj \) or \( vj \) for \( j \in [1, k] \) are frozen since they belong to unbalanced triangles and thus this would contradict Observation 2.1 and Lemma 2.2.

**First item.** Let \( j \) be an integer such that \( j > i \). After pivoting at \( i \), \( x'(e) = x(iu) + x(iv) \), so we want to prove that \( |x'(uj) - x'(vij)| \leq x(iu) + x(iv) \) and \( x(iu) + x(iv) \leq x'(uj) + x'(vij) \).

We start with the latter inequality. From the ordering and the fact that \( j > i \), we know that \( x(iu) + x(iv) \leq x(uj) + x(vj) \). Therefore, for the inequality to be violated, one of \( uj \) or \( vj \) must have been reduced. Say that it is \( uj \), thus we have \( x'(uj) = x(iu) + x(ij) \). Since we also have \( x'(vij) \geq x(vi) - x(ij) \), we obtain \( x'(uj) + x'(vij) \geq x(uj) + x(iv) \) and the inequality is actually not violated.

For the former inequality, let us assume without loss of generality that \( x'(uj) - x'(vij) > x(uj) + x(iv) \), the other case being symmetric. Then we have \( x'(uj) > x(uj) + x(iv) + x'(vij) \), and thus it cannot be that we have both \( x'(uj) \leq x(uj) + x(ij) \) and \( x(ij) \leq x(iv) + x'(vij) \). Therefore at least one of the two triangles \( uij \) or \( ivj \) is unbalanced after pivoting at \( i \), which is impossible by Observation 2.1.

**Second item.** Let \( j \) be an integer such that \( j > i \). After pivoting at \( i \), up to symmetry we can assume that \( x'(e) = x(iu) - x(iv) \), so we want to prove that \( |x'(uj) - x'(vij)| \leq x(iu) - x(iv) \) and \( x(iu) - x(iv) \leq x'(uj) + x'(vij) \).

We start with the former inequality, and first prove that \( x'(uj) - x'(vij) \leq x(iu) - x(iv) \). From the ordering and the fact that \( j > i \), we know that \( x(iu) - x(iv) \geq x(uj) - x(vij) \). Therefore, for the inequality to be violated, the weight of \( uj \) must have increased or the weight of \( vj \) must have decreased. If the weight of \( vj \) decreased, \( x'(vij) = x(ij) + x(iv) \), but then with \( x'(uj) \leq x(ij) + x(uu) \) we obtain \( x'(uj) - x'(vij) \leq x(uu) - x(iv) \), a contradiction. If the weight of \( uj \) increased, \( x'(uj) = |x(iu) - x(ij)| \), which we combine with \( x'(vij) \geq x(ij) + x(vi) \) to obtain \( x'(uj) - x'(vij) \leq x(uu) - x(vi) \). The proof that \( x'(vij) - x'(uj) \leq x(uu) - x(iv) \) is similar: if the weight of \( uj \) decreased, \( x'(uj) = x(ij) + x(uu) \), which combined with \( x'(vij) \leq x(ij) + x(vi) \) gives \( x'(vij) - x'(uj) \leq x(vi) - x(uu) \leq x(vi) - x(uu) \).

For the latter inequality, if it is violated we have \( x'(vij) > x(vi) + x'(uj) + x'(vij) \). As in the first item, this means that one of the two triangles \( uij \) or \( ivj \) is unbalanced after pivoting, which is impossible by Observation 2.1.

In order to prove Theorem 1.1, we use a lower bound on the value of the optimal solution which relies on linear programming duality. A hitting set for \( T' \) is a set of edges \( H \) so that each triangle of \( T' \) contains at least one edge of \( H \). A fractional packing for a set of triangles \( T' \) is a set of values \( (p_t)_{t \in T} \) so that for each edge \( e \), we have \( \sum_{t \ni e} p_t \leq 1 \). The value of the fractional packing is \( \sum_{t \in T'} p_t \).

**Lemma 2.4:** Any solution of **Metric Violation Distance** is at least the value of the smallest hitting set for \( T' \), which is itself lower bounded by the maximal value of a fractional packing of triangles in \( T' \).

**Proof:** Any solution of **Metric Violation Distance** must hit at least one edge of each unbalanced triangle, since otherwise that triangle remains unbalanced and thus we do not obtain a valid metric. The linear programming relaxation of the hitting set problem dualizes to the problem of finding a maximal fractional packing. Therefore the value of the maximal fractional packing is a lower bound for the solution of **Metric Violation Distance**.

The value of the (non-fractional) smallest hitting set of \( T' \) might be different from **Metric Violation Distance**.
even though such a hitting set \( H \) is hitting all the unbalanced triangles, there might be no way of replacing the value of the weight of an edge in \( H \) without recreating an unbalanced triangle, an example is given in Figure 1.

We are now ready to prove Theorem 1.1.

**Proof of Theorem 1.1:** We first observe that the solution output by Algorithm 1 indeed forms a metric. By Observation 2.1, at each round the algorithm repairs the triangles incident to the pivot, and by Lemma 2.2, these triangles stay repaired as the algorithm progresses. Since at the end of the algorithm, every vertex has been chosen once as a pivot, all the triangles are repaired and thus the triangle inequality is satisfied everywhere.

We denote by ALG the output of the algorithm and by OPT the optimal solution and recall that \( T \) denotes the set of all triples and \( T' \) the set of all unbalanced triangles with respect to the input distances. For a triangle \( ijk \in \binom{\mathcal{V}}{3} \), let \( A_{ijk} \) be the indicator of the event that one of them is chosen as a pivot and one of the edges (i.e., the edge not incident to the pivot) was modified as a result. Note that in the execution of the algorithm one edge can be modified many times but this event happens at most once for each triangle since after pivoting at a vertex its adjacent edges are frozen. Let \( p_{ijk} = \mathbb{E}[A_{ijk}] \), where the expectation is taken over the algorithm’s randomness. Then \( \mathbb{E}[ALG] \leq \sum_{t \in T} p_t \). By Lemma 2.2, a triangle never becomes unbalanced in the course of the algorithm, therefore \( p_t = 0 \) for \( t \notin T' \). Thus \( \mathbb{E}[ALG] \leq \sum_{t \in T'} p_t \).

To show that the algorithm is an \( \alpha \)-approximation, we prove that for every edge \( e \), \( q_e := \sum_{t \in T', t \ni e} p_t \leq \alpha \). This shows that \( p_{e} \alpha \) is a fractional packing for the unbalanced triangles, and therefore, by Lemma 2.4, we have

\[
\sum_{t \in T'} \frac{p_t}{\alpha} \leq \text{OPT},
\]

and thus

\[
\mathbb{E}[ALG] \leq \sum_{t \in T'} p_t \leq \alpha \text{OPT}.
\]

Hence, for the rest of the proof, we show that \( q_e = O(\log n) \) for every \( e \). Fix an edge \( e \) that belongs to at least one unbalanced triangle. Note that, for any unbalanced triangle \( t \), when the event indicated by \( A_t \) happens, one of the three vertices of \( t \) gets chosen uniformly at random, and the opposite edge gets modified. Therefore, we have that \( \sum_{t \in T', t \ni e} p_t / 3 = \mathbb{E}[\#\text{times that } e \text{ is modified}] \), and thus it suffices to bound the latter expectation.

We will induct on the number \( n' \) of unbalanced triangles that \( e \) belongs to. By Lemma 2.2, this number is non-increasing throughout the algorithm. Let \( c_{e}(n') \) be an upper bound on the expected number of modifications when \( e \) is adjacent to \( n' \) unbalanced triangles. For \( n_1 \) and \( n_2 \) such that \( n_1 + n_2 = n' \), let us denote by \( E_{n_1, n_2} \) the event that a pivot is chosen among the vertices forming an unbalanced triangle with \( e \), and among the \( n' \) unbalanced triangles, pivoting at \( n_1 \) of them would induce an increase in the weight of \( e \) and pivoting at \( n_2 \) of them would induce a decrease of the weight of \( e \). Note that among pairs \((n_1, n_2)\) such that \( n_1 + n_2 = n' \), at most one \( E_{n_1, n_2} \) holds.

Conditioned to \( E_{n_1, n_2} \), for \( 1 \leq k \leq n_1 \) and \( 1 \leq k' \leq n_2 \), \( F_{k, n_2} \) and \( G_{n_1, k'} \) denote respectively the events that the pivot we choose induces the \( k \)th biggest increase, respectively the \( k' \)th biggest decrease (ties broken arbitrarily). Since pivots are chosen randomly, all these events are disjoint and happen with probability \( 1/n' \). By Lemma 2.3, after the event \( F_{k, n_2} \), at most \( n_2 + k - 1 \) unbalanced triangles remain. Likewise, after the event \( G_{n_1, k'} \), at most \( n_1 + k - 1 \) unbalanced triangles remain.

We prove by induction that \( c_{e}(n') \leq c \ln(n' + 1) \) for some constant \( c > 2 \) to be determined. For the base case, we use the easy bound \( c_e(n') \leq n' \leq c \ln(n' + 1) \) for any \( n' \leq c \).

\[
c_e(n') \leq 1 + \sum_i \mathbb{P} \left[ i \text{ had bad triangles remain after pivoting} | c_e(i) \right] \leq 1 + \sum_{n_1 + n_2 = n'} \mathbb{P} \left[ E_{n_1, n_2} \right] \mathbb{P} \left[ F_{k, n_2} \right] c_e(k - 1 + n_2) + \sum_{k' = 1}^{n_2} \mathbb{P} \left[ G_{n_1, k'} \right] c_e(k' - 1 + n_1) \leq 1 + \max_{n_1 + n_2 = n'} \left( \sum_{k = 1}^{n_1} \mathbb{P} \left[ F_{k, n_2} \right] c_e(k - 1 + n_2) + \sum_{k' = 1}^{n_2} \mathbb{P} \left[ G_{n_1, k'} \right] c_e(k' - 1 + n_1) \right) \leq 1 + \max_{n_1 + n_2 = n'} \left( \sum_{k = 1}^{n_1} \frac{1}{n'} c_e(k - 1 + n_2) + \sum_{k' = 1}^{n_2} \frac{1}{n'} c_e(k' - 1 + n_1) \right) \leq 1 + \max_{n_1 + n_2 = n'} \left( \sum_{k = n_1+1}^{n_1'} \frac{1}{n'} c_e(k - 1) + \sum_{k' = n_2+1}^{n_2'} \frac{1}{n'} c_e(k' - 1) \right) \leq 1 + \left( \sum_{k = [n'/2] + 1}^{n'} \frac{1}{n'} c_e(k - 1) \right) + \left( \sum_{k' = [n'/2] + 1}^{n'} \frac{1}{n'} c_e(k' - 1) \right) \leq 1 + \frac{1}{n'} c \ln \left( \frac{\left( n' \right)!}{\sqrt{2\pi n'}} \right) \leq c \ln(n' + 1).
\]

The inequality (1) comes from the fact that the sum is maximized when \( n_1 = [n'/2] \), (2) is the induction hypothesis, and (3) follows from a Stirling approximation \( n^{n+1}2^{n}e^{-n} \leq n! \leq cn^{n+1}e^{-n} \) (see full version [34] for more details).

The example in Figure 2 shows that the analysis in Theorem 1.1 is tight.
Lemma 2.5: Let \( x \) denote the distance data described by Figure 2. Then in expectation, Algorithm 1 yields a solution of cost \( \Omega(\log n) \), while the optimal solution has cost 1.

Proof: It is straightforward to see that the optimal solution has cost 1: it has cost at least one since there is (at least) one unbalanced triangle, and changing the weight of the \( vw \) edge to a value in \([0, 2]\) yields a valid metric.

For the upper bound, we first observe that the unbalanced triangles in this instance are exactly the triangles \( vwu_k \). The effect of Algorithm 1 will be to repair these triangles in some order.

If the first pivot is at a vertex \( u_k \), the effect of the pivot on the weights is to change the weight of \( vw \) to \( 2k \). After this change, the triangles \( vwu_j \) for \( j > k \) are no longer unbalanced, while the triangles \( vwu_j \) for \( j < k \) are still unbalanced. Future pivots at a vertex \( u_{k'} \neq v, w \) work similarly and repair exactly the triangles \( vwu_j \) for \( j > k' \).

If at any point, \( v \) or \( w \) is chosen as a pivot, each unbalanced triangle \( vwu_k \) gets repaired by changing the weight of \( vwu_k \) or \( wu_k \) to (roughly) \( 2k \), and then the instance is fully repaired.

Therefore, the number of edges changed by Algorithm 1 is exactly \( k \), where \( k \) is the smallest index of a vertex \( u_k \) that was chosen as a pivot before \( v \) or \( w \) was chosen. Since pivots are chosen uniformly at random, for \( 1 \leq m \leq n + 2 \), the probability that \( v \) or \( w \) gets chosen as the \( m \)th pivot is \( \Theta(1/n) \), and the minimum index of pivots chosen until then is \( \Theta(n/m) \) in expectation. Therefore the expected value output by the algorithm is \( \Theta(\sum_m 1/n \times n/m) = \Theta(\log n) \).

III. CONSTANT-FACTOR APPROXIMATION FOR ULTRAMETRIC VIOLATION DISTANCE

In this section, we provide an algorithm for the ultrametric setting and prove Theorem 1.2. We first recall some basic properties regarding ultrametrics. We refer to [10], [17] for more detailed reviews of ultrametric properties.

A. Preliminaries on ultrametrics

An ultrametric is a metric \((X, \text{dist})\), where \( X \) is a set of points and \( \text{dist} : X \times X \to \mathbb{R}_+ \) is such that \( \forall i, j, k \in X \), \( \text{dist}(i, j) \leq \max(\text{dist}(i, k), \text{dist}(j, k)) \). Equivalently, an ultrametric can be represented by a rooted edge-weighted tree whose sets of leaves is \( X \) and where the leaf-to-root distance is the same for all leaves, i.e.: all the elements in \( X \). Then, the distance between any pair of elements is given by the distance in the edge-weighted tree.

The tree representation \( T \) of a given ultrametric is pretty useful for algorithmic purposes. For each distance \( d \), one can consider the clustering induced by \( d \), namely the connected components of the tree obtained from \( T \) by removing the nodes at distance larger than \( d/2 \) from the leaves. In this case, all the leaves that are in the same connected component are at pairwise distance at most \( d \) and no pair of leaves at distance smaller than \( d \) are in different connected components.

B. Our Algorithm

To simplify the exposition, we take a graph perspective on the problem: Given an instance of ULTRAMETRIC VIOLATION DISTANCE \( x \in \mathbb{R}^{\binom{n}{2}} \), we define a weighted graph whose vertex set is \([n]\) and the edge distances are given by the distances in \( x \). We will henceforth call the elements of \([n]\) vertices and refer to an instance of ULTRAMETRIC VIOLATION DISTANCE as a graph.

Throughout this section, we will use the correlation clustering problem, as a subproblem to solve to obtain a constant factor approximation for the ultrametric setting. Our algorithm defines complete graphs where each edge is labeled with a + or a - sign. In this context, the correlation clustering problem asks for a partition of the vertices minimizing the number of -edges that are fully contained within a cluster plus the number of + edges across clusters. Our algorithm is recursive and goes as follows.

Given an instance of ULTRAMETRIC VIOLATION DISTANCE \( x \in \mathbb{R}^{\binom{n}{2}} \), let \( w_1 < \ldots < w_L \) be the distinct distance values appearing in \( x \). The algorithm first creates the following correlation clustering instance: consider the weighted graph induced by the instance \( x \) as explained above. Then, each edge of weight \( w_L \) is replaced with a - edge and all the other edges, namely of weight less than \( w_L \), are + edges. The algorithm then computes an approximate solution to the above correlation clustering instance using an algorithm presented in the full version [34, Section 3.1], called agreement correlation clustering, and whose properties are captured by Theorem 3.2 below. The agreement correlation clustering algorithm produces a partition of the vertices. Then, our algorithm fixes the distances between any pair in different clusters to be \( w_L \), and creates a subinstance of the ULTRAMETRIC VIOLATION
DISTANCE problem for each cluster by setting the distances between vertices in the same clusters to be the minimum between their original distance and \( w_{L-1} \). The algorithm then makes a recursive call within each cluster. We give a full description of the algorithm on Algorithm 2.

**Algorithm 2**: Procedure \( \mathcal{A} \): An \( O(1) \)-approximation for the ultrametric violation distance

1. **Input**: An \( n \times n \) dissimilarity matrix \( x \);
2. **Output**: \( \text{dist} \): Distances between all the elements of \( x \), such that \( \text{dist} \) is an ultrametric;
3. \( w_{\text{max}} \leftarrow \) maximum distance entry in \( x \);
4. \( \mathcal{C}(w_{\text{max}}) \leftarrow \) Correlation clustering instance over the elements of \( x \) where each pair \( u, v \) such that the weight in \( x \) is less than \( w_{\text{max}} \) is a + edge, each pair \( u, v \) such that the weight is \( w_{\text{max}} \) is a - edge;
5. \( \mathcal{C}(w_{\text{max}}) \leftarrow \) Solution to \( \mathcal{C}(w_{\text{max}}) \) obtained by agreement correlation clustering (see Theorem 3.2) on \( \mathcal{C}(w_{\text{max}}) \);
6. For any pair of \( u, v \) such that \( u \in C_i, \ v \in C_j, \ i \neq j \), \( \text{dist}(u, v) \leftarrow w_{\text{max}} \);
7. For any pair of \( u, v \) such that \( u, v \in C_i \), let \( x(u, v) \leftarrow \min(x(u, v), w_{\text{max}}) \);
8. **foreach** \( i = 1, \ldots, k \) **do**
9. **if** \( \lvert C_i \rvert > 1 \) **then**
10. Compute the distances between the elements of \( C_i \) by making a call to \( \mathcal{A} \) on \( x \) restricted to the elements of \( C_i \);
11. **else**
12. \( \text{dist}(u, u) \leftarrow 0 \) where \( u \) is the unique element of \( C_i \);

The following fact follows from the definition of the algorithm and the observation that given an ULTRAMETRIC VIOLATION DISTANCE instance with maximum distance \( w \), the algorithm returns an ultrametric with maximum distance \( w \).

**Lemma 3.1**: The above algorithm produces an ultrametric in polynomial time.

**Proof**: The running time of the algorithm follows immediately from its definition and Theorem 3.2.

We now argue that for any \( i, j, k \), we have that the so-called "triple condition": \( \text{dist}(i, j) \leq \max\{\text{dist}(i, k), \text{dist}(k, j)\} \). Observe first that the distances defined by the algorithm are monotonically decreasing over the recursive calls: Namely, at each execution of \( \mathcal{A} \) where the maximum distance entry in \( x \) is \( w_{\text{max}} \), we have that the next recursive calls are such that the maximum distance is strictly smaller than \( w_{\text{max}} \). Thus, consider the first recursive call after which \( i, j, k \) are not in the same cluster produced by the agreement correlation clustering algorithm and let \( w_{\text{max}} \) be the maximum distance in \( x \) at this recursive call. If \( i, j, k \) are all in different clusters, then the distance between them is set to be \( w_{\text{max}} \) and in which case we have \( \text{dist}(i, j) = \text{dist}(j, k) = \text{dist}(i, j) \) and they satisfy the triple condition. Next, assume without loss of generality, that \( i, j \) are in the same cluster and \( k \) is in a different cluster. In this case, \( \text{dist}(i, k) = \text{dist}(j, k) = w_{\text{max}} \). Moreover, as argued above, we have that \( i, j \) are for the first time in different clusters at following recursive calls and so \( \text{dist}(i, j) < w_{\text{max}} \). We thus have that \( i, j, k \) satisfy the triple condition and dist is indeed an ultrametric.

Let \( \epsilon > 0 \) be a constant satisfying
- \( \frac{1}{1 - 1/3} > \epsilon / 8 \), and
- \( \frac{1}{1 - 1/4} > \epsilon / 8 \).

We can pick \( \epsilon < 1/50 \) to satisfy the above constraints.

We will use the following crucial notion. Given a correlation clustering instance, we say that a set of vertices \( C \) is important if for any vertex \( v \in C \), \( v \) has at most an \( \epsilon / 8 \) fraction of its + neighbors outside \( C \) and is + connected to at least a \( (1 - \epsilon / 8) \) fraction of the vertices in \( C \). These groups of vertices are in essence dense + regions. Moreover, for a given set of vertices \( C \), we say that \( C \) is everywhere dense if for any \( v \in C \), \( v \) has a + edge to at least \( 2|C|/3 \) vertices of \( C \). We say that a singleton is everywhere dense.

We will make use of the following theorem proved in the full version [34]. The structure that it guarantees is illustrated in Figure 3.

**Theorem 3.2**: Let \( S = \{S_1, \ldots, S_k\} \) be the set of clusters output by the agreement correlation clustering algorithm. Then, for any important group of vertices \( C \), there is a cluster \( S_i \) such that \( C \subseteq S_i \), and \( S_i \) does not intersect any other important groups of vertices disjoint from \( C \). Moreover, any cluster \( S_i \in S \) is everywhere dense.

Assuming the above theorem we can turn to the proof of the theorem for the ultrametric case.

**Proof of Theorem 1.2**: Given a solution to the ultrametric problem, we define the clustering at level \( i \), to be the partition of the vertices obtained by putting in the same part, the vertices at pairwise distance less than \( w_i \). A solution to the ultrametric problem hence defines nested clusterings. Thus, consider the following set of nested clusterings induced by an optimum solution to the ultrametric problem. The clustering \( C_{L+1} \) at level \( L+1 \) simply consists of one cluster containing the whole vertex set. Then, for each \( w_i \), the optimal solution defines a clustering \( C_i \) that places the vertices that are at distance less than \( w_i \) in the solution in the same clusters.

Furthermore, for each level \( i \), we will consider the correlation clustering instance \( M_i \) where for each pair \( (u, v) \), there
is a - edge if the input distance between \((u, v)\) is \(w_i\) or higher and a + edge otherwise.

We then apply the following top-down transformation to an optimum solution \(OPT\) to the ULTRAMETRIC VIOLATION DISTANCE problem so as to obtain a solution \(OPT'\) with more structure and whose cost is within a constant factor of the cost of \(OPT\). Consider the nested clustering induced by \(OPT\) as defined above and a cluster \(C\) at some level \(i\) such that either the number of - edges between pairs of vertices of \(C\) in \(M_i\) is at least \(\varepsilon^2 |C|^2/800\) or the cut \(C, V - C\) has more than \(\varepsilon^2 |C|^2/800 +\) edges. Then, we simply set the distance between any pair of vertices of \(C\) to be \(w_i\) (which implies that the clustering induced by \(OPT'\) on \(C\) at each level \(j \leq i\) consists of singleton elements of \(C\)). We will refer to this operation as making singleton clusters for levels \(i\) and below.

Next, we proceed top-down on the non-singleton clusters of the nested clustering resulting from the above transformation. For each cluster \(C\) at level \(i\), as long as there is a vertex \(v\) that has either more than an \(\varepsilon/8\) fraction of its + neighbors outside \(C\), or less than \((1 - \varepsilon/8)|C| +\) neighbors within \(C\), set the distance from \(v\) to the elements of \(C\) to be \(w_i\). This implies that in the resulting solution, \(v\) is in a singleton cluster for any level \(j \leq i\). Again, we say in this case that \(v\) is made a singleton cluster for levels \(i\) and below.

This completes the description of \(OPT'\). We have the following lemma, whose proof is in the full version [34]

**Lemma 3.3:** The cost of \(OPT'\) is at most \(800/\varepsilon^2\) times the cost of \(OPT\). Moreover, at each level, the clusters induced by \(OPT'\) are either singletons or important groups of vertices. For a given level \(i\), we let the important clusters of \(OPT'\) be the non-singleton clusters of \(OPT'\) at this level. Note that the important clusters are important groups of vertices that are not split in \(OPT'\), i.e., in the same cluster of \(OPT'\).

Furthermore, given any solution \(S\), we say that a pair of vertices \((u, v)\) is separated at level \(i\) if the distance between \(u, v\) in the solution is \(w_i\). This means that the nested clusterings induced by solution \(S\) are such that \(u, v\) are in the same cluster for any clustering of level \(j > i\) and in different clusters in the clusterings at level \(j \leq i\).

We now show the following lemma, whose proof is in the full version [34]

**Lemma 3.4:** Consider a solution \(S\) for the ULTRAMETRIC VIOLATION DISTANCE problem such that for any level \(i\), the clustering \(C_i\) induced by the solution at level \(i\) satisfies the following properties,

1. Each cluster \(C \in C_i\) is everywhere dense;
2. For each cluster \(C \in C_i\) there is at most one important cluster \(C'\) of \(OPT'\) at level \(i\) that intersects \(C\);
3. Each important cluster \(C'\) of \(OPT'\) at level \(i\) is fully contained in a cluster of \(C_i\).

Then the cost of solution \(S\) is at most 4 times the cost of solution \(OPT'\).

Equipped with this, we now prove that the algorithm of the previous section produces a good solution by showing that the solution output is such that for any level \(i\), the clustering \(C_i\) induced by the solution at level \(i\), satisfies the following properties,

1. Each cluster \(C \in C_i\) is everywhere dense;
2. For each cluster \(C \in C_i\) there is at most one important cluster \(C'\) of \(OPT'\) at level \(i\) that intersects \(C\);
3. Each important cluster \(C'\) of \(OPT'\) at level \(i\) is fully contained in a cluster of \(C_i\).

We are now ready to conclude the proof of the theorem. We proceed by induction on the level to show that any cluster produced by the algorithm is everywhere dense, each important cluster of \(OPT'\) is contained in some cluster produced by the algorithm, and each cluster output intersects at most one important cluster of \(OPT'\). At the top level, these properties trivially hold since everything is a single cluster both in \(OPT'\) and in \(S\).

Now, consider an important cluster \(C\) of \(OPT'\) at level \(i\), then by definition of \(OPT'\) it is a subset of an important cluster \(C'\) of \(OPT'\) at level \(i + 1\). By induction hypothesis, there is a cluster \(S'\) of the algorithm at level \(i + 1\) containing \(C'\), we thus have \(C \subseteq C' \subseteq S'\). Since \(C\) is an important group of vertices for the entire clustering instance \(M_i\) and \(C' \subseteq S'\), it is an important group of vertices for the subinstance of \(M_i\) induced by the vertices in \(S'\) and so by Theorem 3.2, there is a cluster \(S\) of the clustering output by the algorithm at level \(i\) containing \(C\) and does not intersect any other important cluster \(C''\) of \(OPT'\) (because such \(C''\) also remains important in the subinstance). Finally, Theorem 3.2 also ensures that each cluster is everywhere dense. Therefore, we can apply Lemma 3.4 to \(S\) and invoke Lemma 3.3 to conclude that \(S\) is a 3200/\(\varepsilon^2\)-approximation to the ultrametric problem.

**ACKNOWLEDGMENT**

We are grateful to Alantha Newman for helpful discussions.

**REFERENCES**

[1] C. Elkam, “Using the triangle inequality to accelerate k-means,” in Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ser. ICML’03, 2003, pp. 147–153.

[2] M. O. Dayhoff and R. M. Schwartz, “Chapter 22: A model of evolutionary change in proteins,” in Atlas of Protein Sequence and Structure, 1978.

[3] C. Elkan, “Using the triangle inequality to accelerate k-means,” in Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ser. ICML’03, 2003, pp. 147–153.

[4] J. Brickell, I. S. Dhillon, S. Sra, and J. A. Tropp, “The metric nearness problem,” SIAM Journal on Matrix Analysis and Applications, vol. 30, no. 1, pp. 375–396, 2008.

[5] C. Fan, B. Raichel, and G. Van Buskirk, “Metric violation distance: Hardness and approximation,” in Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms. SIAM, 2018, pp. 196–209.

[6] C. Fan, A. C. Gilbert, B. Raichel, R. Sonthalia, and G. Van Buskirk, “Generalized metric repair on graphs,” in 17th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[7] G. Duggal, R. Patro, E. Sefer, H. Wang, D. Filippova, S. Khuller, and C. Kingsford, “Resolving spatial inconsistencies in chromosome conformation measurements,” Algorithms for Molecular Biology, vol. 8, no. 1, pp. 1–10, 2013.

[8] V. V. Williams and R. R. Williams, “Subcubic equivalences between path, matrix, and triangle problems,” Journal of the ACM (JACM), vol. 65, no. 5, pp. 1–38, 2018.
