On constant factor approximation for earth mover distance over doubling metrics

Shi Li
Dept. of Computer Science
Princeton University
35 Olden Street
Princeton, NJ, 08540
shili@cs.princeton.edu

February 9, 2010

Abstract

Given a metric space \((X,d_X)\), the earth mover distance between two distributions over \(X\) is defined as the minimum cost of a bipartite matching between the two distributions. The doubling dimension of a metric \((X,d_X)\) is the smallest value \(\alpha\) such that every ball in \(X\) can be covered by \(2^\alpha\) ball of half the radius. A metric (or a sequence of metrics) is called doubling precisely if its doubling dimension is bounded.

Our first result is a near linear time (in the size of the \(X\)) algorithm for estimating EMD over doubling metric \((X,d_X)\), with an approximation ratio \(O(\alpha_X)\) preprocessing time, creating a data structure of size \(\tilde{O}(n^{1+\epsilon})\), such that subsequent EMD queries can be answered in \(\tilde{O}(n)\) time, with approximation ratio \(O(\alpha_X/\epsilon)\).

Our second result is an encoding scheme, which is a weaker form of sketching. In an encoding scheme, distributions are encoded, such that the EMD between two distributions can be estimated in sub linear time, given the encodings of the two distributions. In particular, given \((X,d_X)\), by using \(\tilde{O}(n^2)\) preprocessing time, every subsequent distribution \(\mu\) can be encoded into \(F(\mu)\) in \(\tilde{O}(n^{1+\epsilon})\) time. The query for EMD between \(\mu\) and \(\nu\) can be answered in \(\tilde{O}(n')\) time, with approximation ratio \(O(\alpha_X/\epsilon)\), given the two encodings \(F(\mu)\) and \(F(\nu)\).

The encoding scheme has immediate applications. In a 2-player game where 1 player knows \(\mu\) and the other knows \(\nu\), there is a communication protocol with small communication complexity, through which the two players can approximate the EMD between \(\mu\) and \(\nu\). Another application is distance oracle, where we are given a metric \((X,d_X)\) and \(s\) distributions \(\mu_1,\mu_2,\ldots,\mu_s\) over \(X\), we can use \(\tilde{O}(n^2+sn^{1+\epsilon})\) preprocessing time, creating a data structure of size \(\tilde{O}(n^{1+\epsilon}+sn)\), such that query for EMD between \(\mu_i\) and \(\mu_j\) can be answered in \(\tilde{O}(n')\) time, with approximation ratio \(O(\alpha_X/\epsilon)\).

1 Introduction

Given a finite metric \((X,d_X)\) and two multi-sets \(A,B\) of points in \(X\) with \(|A|=|B|=N\), the Earth-Mover distance (or EMD) between \(A\) and \(B\) is defined as the minimum cost of a perfect matching, with respect to the cost function \(d_X\), i.e:

\[
EMD_X(A,B) = \min_{\pi: A\rightarrow B} \sum_{a \in A} d_X(a, \pi(a))
\]

where \(\pi\) ranges over all one-to-one mappings.

The EMD metric is of significant importance in many applications. For example, in computer vision, EMD is used as a measurement for dissimilarity between two images. The idea is to represent an image as a distribution on features (such as colors and color spectrum), with an underlying metric. Then, the EMD of two distributions of features can be computed in \(O(N^3)\) time for general underlying metrics, using Hungarian method [18]. For metric supported by a sparse graph, there is a scaling algorithm due to [12] that runs in \(O(\sqrt{|V||E| \log(|V||N|)})\) time. Better algorithms are known for special metrics. EMD over a 2-dimensional plane can be computed in
Even for the special metric such as 2-dimensional plane, the exact EMD requires super-quadratic time, too expensive in many applications. This computational bottleneck motivates faster approximation algorithms. For the EMD over 2-dimensional plane, Agarwal and Varadarajan [24] showed \( O(N^{3/2}/\epsilon^2 \log^5(N/\epsilon)) \) time \((1 + \epsilon)\)-approximation, then they gave an improved algorithm with \( N^{1+\delta}\log^{O(1)}N \)-time and \( \log(1/\delta)\)-approximation[2]. Indyk [14] further improved the running time to \( O(N \log^{O(1)} N) \) for constant approximation. Following [14], Andoni etc. [4] provided a sketching algorithm with constant approximation.

There has also been special interest for embedding EMD into normed spaces. Charikar [8] showed that EMD over \((X, d_X)\) can be embedded into \( l_1 \) space with distortion \( \alpha \), if \((X, d_X)\) can be embedded into distribution of dominating trees with distortion \( \alpha \). Then, by [11], which showed \( \alpha \) is at most \( O(\log n) \) for a \( n \)-point metric, EMD over \((X, d_X)\) with can be embedded into \( l_1 \) with distortion \( O(\log n) \). Embedding into \( l_1 \) can give approximation algorithms, with the ratio equal to the distortion of the embedding. However, the embedding of EMD into \( l_1 \) has limitations: it has been showed in [19] that embedding of EMD over grid \( [n]^2 \) must incur a distortion of at least \( \Omega(\sqrt{\log n}) \).

In this paper, we are interested in approximating EMD over more general metrics: metrics with bounded doubling dimensions. The doubling dimension of a metric \((X, d_X)\) is the smallest \( k \) such that every ball in \( X \) can be covered by \( 2^k \) balls of half the radius. This is a richer family of metrics than the constant-dimension Euclidean space. It can be shown, for example, for any fixed \( p \), the doubling dimension of \( d \)-dimensional \( l_p \) is roughly \( d \). On the other hand, [13] showed a family of metrics \((G_k, d_k)\) with bounded doubling dimension, whose embedding into \( l_2 \) must incur a distortion of \( \Omega(\sqrt{|G_k|}) \). For a finite metric space \( X \), \( \alpha_X \leq \log |X| \). For many problems involving metrics, better results are known if the metrics have bounded doubling dimension. [10] obtained an \( O(\sqrt{\alpha_X \log n}) \) distortion embedding of \( X \) into \( l_2 \), an improvement over Bourgain’s theorem[1] if the metric has low doubling dimension. [5] give better approximation algorithms for metric labeling and \( 0 \)-extension for \( \alpha \)-decomposable metrics (if a metric has doubling dimension \( \alpha \), it is \( O(\alpha) \)-decomposable).

\[ O(N^{5/2} \log^{O(1)} N) \] time, due to Vaidya[23]. This time is further improved to \( O(N^{2+\delta}) \), for any \( \delta > 0 \), by Agarwal etc. [3].

### 1.1 Our contribution

We study approximation algorithm for EMD over doubling metrics, a generalization of low dimensional Euclidean spaces. As far as we know, this is the first work to consider the EMD over this family of metrics. Our algorithm is a generalization of the algorithm in [14] for approximating EMD over planar grids. We defined a metric, called “sibling-linked hierarchical well-separated tree” metric, that the doubling metric can be embedded into with constant distortion. This metric has a nice property that the EMD over it is the sum of EMD over smaller metrics, which allows us to do “importance sampling”. In this paper, we also promoted a scheme called “encoding scheme”, a weaker form of the sketching scheme. In the encoding scheme, distributions are encoded in some form, and the EMD of two distributions can be approximated in sub-linear time, if the encodings of the two distributions are given. The sub-linear time estimation algorithm does the importance sampling in a binary-search way, using only logarithmic time.

Compared to the Euclidean spaces, there is an issue for doubling metrics on how the underlying metric is given. Reading the whole metric requires time quadratic in \( n \), the size of the metric, while we’re aiming for algorithms with time near linear in \( n \). We avoid this bottleneck by preprocessing, in which our algorithms read the metric \((X, d_X)\) and create a data structure of small size. By doing this, subsequent queries for EMD between \( \mu \) and \( \nu \) can be answered in time near linear in \( n \).

Throughout, we use \( \alpha_X \) to denote the doubling dimension of \( X \). We assume each coordinate in a distribution can be represented in polylog\( (n) \) bits. In particular, for a fixed constant number \( a \), we define

\[
\mathcal{P}_X = \left\{ \mu : X \to \left\{ 0, \frac{1}{n^a}, \frac{2}{n^a}, \ldots, 1 \right\} \middle| \sum_{p \in X} \mu_p = 1 \right\}
\]

This restriction is only for simplicity of demonstration and doesn’t affect our algorithms.

Our first result is an almost linear time, constant approximation algorithm for EMD over doubling metrics.

**Theorem 1.1** (Approximation algorithm). Let \((X, d_X)\) be a metric space with \( |X| = n \). Let \( 0 < \epsilon < 1 \) be fixed. Given \((X, d_X)\) and \( \alpha_X \), there is an algorithm which, by using \( O(n^2) \) preprocessing time to create a data structure of size \( O(n^{1+\epsilon}) \), for any subsequent query for EMD between two distributions \( \mu, \nu \in \mathcal{P}_X \), outputs in \( O(n) \) time a random estimation \( D \) satisfy-
EMD_X(\mu, \nu) \leq D \leq O\left(\frac{\alpha_X}{\epsilon}\right) EMD_X(\mu, \nu) \tag{1}

with probability at least 2/3. The probability is over the randomness for the preprocessing as well as the estimation.

The preprocessing time is quadratic in the size of the metric, which is unavoidable since we have to read the whole metric. However, the algorithm only reads the metric once, after which subsequent EMD queries can be answered in almost linear time in n. The probability 2/3 can be amplified to 1 - \epsilon, by repeating the algorithm \Omega(\log(1/\epsilon)) times.

Unlike [4], our algorithm for theorem 1.1 is not sketching-based. Instead, we provide a weaker scheme, which we call “encoding scheme”. In the encoding scheme, distributions are encoded to l_1 vectors by an encoding function F. Given two encodings F(\mu) and F(\nu), there is a sub linear time algorithm approximating EMD_X(\mu, \nu). The difference between an encoding and a sketch is, an encoding is not necessarily shorter than its pre-image.

**Theorem 1.2** (Encoding scheme). Let X, d_X, n, \epsilon be as in theorem 1.1. Given (X, d_X) and \alpha_X, there is an algorithm, which uses \tilde{O}(n^2) preprocessing time to create a data structure of size \tilde{O}(n^{1+\epsilon}), such that for any subsequent query for EMD_X(\mu, \nu), it can perform the following 3 steps:

1. computes an encoding F(\mu) of size \tilde{O}(n) for \mu in \tilde{O}(n^{1+\epsilon}) time, only reading \mu and the data structure;
2. computes F(\nu) similarly;
3. outputs a random number D satisfying \[\text{1} \] with probability 2/3, in \tilde{O}(n^*) time, only reading F(\mu), F(\nu) and the data structure.

Notice that compared to the algorithm in theorem 1.1, the algorithm in theorem 1.2 requires \tilde{O}(n^{1+\epsilon}) time to approximate the EMD. The encoding scheme implies the following two theorems.

**Theorem 1.3** (Communication protocol). Let X, d_X, n, \epsilon be as in theorem 1.1. Consider a game with two players Alice and Bob, in which Alice knows \mu \in \mathcal{P}_X and Bob knows \nu \in \mathcal{P}_X. There exists a communication protocol with complexity \tilde{O}(n^*), through which Alice and Bob can output a number D satisfying \[\text{1} \] with probability at least 2/3.

Theorem 1.3 suggests that proving a good lower bound for sketching using general communication complexity is impossible.

**Theorem 1.4** (Distance oracle). Let X, d_X, n, \epsilon be stated as in theorem 1.1. Given (X, d_X), \alpha_X and s distributions \mu_1, \mu_2, \cdots, \mu_s \in \mathcal{P}_X, there is an algorithm which uses preprocessing time \tilde{O}(n^2 + s^{1+\epsilon}) to construct a data structure of size \tilde{O}(n^{1+\epsilon} + s), and for any subsequent query for EMD between \mu = \mu_i and \nu = \mu_j, outputs a number D in \tilde{O}(n^*) time, satisfying \[\text{1} \] with probability at least 2/3.

### 1.2 Preliminaries

The doubling dimension \alpha_X of a metric (X, d_X) is defined as the minimum t such that every ball in X can be covered using 2^t balls of half the radius. Define \( R_X = \min_p \max_{q \in X} d_X(p, q) \) to be the radius of X. When X is clear from the context, we may omit the subscript, using \( \alpha, R \) instead.

For a point \( p \in X \) and a real number \( r \), we use Ball(\( p, r \)) = \{ q : d_X(p, q) \leq r \} to denote the ball of radius \( r \) centered at \( p \).

For two distributions \( \mu, \nu \in \mathcal{P}_X \), the earth mover distance (EMD) between \( \mu \) and \( \nu \) is defined as

\[
EMD_X(\mu, \nu) = \min_{\pi : X \times X \rightarrow \mathbb{R}} \sum_{p, q \in X} d_X(p, q)\pi(p, q)
\]

where \( \pi \) ranges over all functions \( \pi \) satisfying

\[
\forall p, q \in X, \pi(p, q) \geq 0; \\
\forall p \in X, \sum_{q \in X} \pi(p, q) = \mu_p; \\
\forall q \in X, \sum_{p \in X} \pi(p, q) = \nu_q.
\]

We’ll use \( l_1 \) to denote the set of \( L_1 \) vectors of finite dimension, i.e \( l_1 = \bigcup_{i=1}^{\infty} \mathbb{R}^i \), equipped with \( L_1 \) norm.

We’ll use \( \oplus \) to denote the direct product operation. So, \( x_1 \oplus x_2 \) is the direct product of \( x_1 \) and \( x_2 \), and \( \oplus_{i=1}^{s} x_i \) is the direct product \( x_1, x_2, \cdots, x_s \).

The Cauchy distribution \( \mathcal{C}(x_0, \gamma) \) is the distribution with the following probability density function : \( f(x) = \frac{\gamma}{\pi} \left( \frac{\gamma}{x - x_0} \right)^2 \), where \( \gamma \) is called the scale parameter of the distribution. The cumulative distribution function of \( \mathcal{C}(x_0, \gamma) \) is \( F(x) = \frac{1}{2} \arctan \left( \frac{x - x_0}{\gamma} \right) + \frac{1}{2} \). Cauchy distribution is 1-stable distribution, meaning that the sum of \( n \) independent variables from \( \mathcal{C}(0, 1) \) is \( \mathcal{C}(0, n) \).

Throughout, we’ll use \( \tilde{O}() \) notion to hide a factor of polylog(n).

### 2 Overview of the algorithms

Our algorithms for theorems 1.1 and theorem 1.2 are only slightly different. We’ll describe the major com-
ponents of the algorithms here, and give the details in following sections.

In the preprocessing, we read \((X, d_X)\) and decompose it into a distribution of so called \textit{sibling-linked hierarchical well separated trees}, or SLHSTs, with distortion \(O(\alpha/\epsilon)\), where \(\epsilon\) is some parameter. The name is a little confusing since SLHST is actually not a tree (recall that even grids can not be embedded into distribution of trees with distortion \(o(\log n)\)). A SLHST is constructed by adding links connecting children of the same vertex (or siblings) to a base HST. In a HST, there is a unique path connecting two leaves. While in SLHST, we take a shortcut for the this path: the two children of the last common ancestor of the two leaves are directly connected. A perfect analogy for this metric is the postal service system. In the postal service system with a hierarchy of post offices, a package is sent from a terminal to a local post office, and then to an even higher level, until it can be sent to an office of the same level. Then the package is sent to lower levels along the hierarchy system, until it reaches the destination.

Our embedding into distribution of SLHSTs is dominating, and has small distortion. Then, by a similar argument as the one in \cite{8}, the EMD over \(X\) can be approximated by the EMD over a random SLHST from the distribution. A nice property that a SLHST has is, the EMD over it can be represented as a sum of \(O(n)\) EMDs over smaller sub metrics. Recall a special case in \cite{14}, the EMD over a grid of size \(n\) is the sum of EMDs over grids of smaller size.

The decomposition into smaller EMDs allows us to use the “importance sampling” technique, which is also used in \cite{13}. Suppose we want to compute a sum \(Z = \sum_{i=1}^{N} Z_i\), but it’s too expensive to compute all the \(Z_i\)s. We can choose a term \(Z_i\) with probability roughly proportional to how much it contributes to the sum. Then \(Z_i\) divided by this probability is used as the estimation for \(Z\). The expectation of the value is \(Z\), while the deviation depends on how well the sampling distribution approximates the weight distribution. Applying importance sampling to our case, we need a technique to approximate each EMD over a small metric up to a reasonable factor. This can be done using the embedding of metrics to distribution of dominating trees and then computing the EMD over the trees (\cite{11, 8}).

The above techniques provided us the main elements needed to prove theorem \cite{13} while not enough to prove theorem \cite{13}. To get a sub-linear time estimation algorithm, we need to do the binary sampling more carefully. We invented a “binary importance sampling” process, which allows us to do the importance sampling in logarithmic time. The high level idea is to sample in a binary-search way. Recall in an importance sampling, we want to compute \(Z = \sum_{i=1}^{N} Z_i\). We maintain a possible set \(S\) of terms, initially \([N]\). Each time the set \(S\) is divided into two equal subsets \(S_1\) and \(S_2\), and then replaced with one of two subsets, with probability proportional to the weight of the subset. In \(\log N\) steps, we end up with a single term in \(S\), which is the term selected by the algorithm. This method requires us to estimate the sum of a subset of terms. In our case, each term can be approximated by the \(L_1\) norm of a vector, so the total weight of a subset is the sum of \(L_1\) norms, which is equivalent to one \(L_1\) norm. We can use the sketch scheme in \cite{17} to approximate the \(L_1\) norm.

For every potentially possible set \(S\), we have a sketch for the concatenation of vectors in \(S\). The sketch is linear, a crucial property allowing us to encode two distributions separately. We’ll describe the encoding function and the binary sampling algorithm in section \cite{6}. 

3 Sibling-linked hierarchical well-separated trees

We introduce the definition of \(k\)-HST from \cite{6} and then based on this definition, we define a \(k\)-SLHST.

\textbf{Definition 3.1} \((k\text{-Hierarchical well-separated tree}(k\text{-HST}))\). A \(k\)-hierarchical well-separated tree \((k\text{-HST})\) is defined as a rooted weighted tree with the following properties:

1. The edge weight from any node to each of its children is the same;
2. The edge weights along any path from the root to a leaf are decreasing by a factor of at least \(k\).

We call \(k\) the scale-decreasing factor of the HST. For a tree \(k\text{-HST} \tau\), we use \(V_\tau, U_\tau, \deg(\tau)\) and \(\text{dep}(\tau)\) to denote \(\tau\)'s vertices, non-leaf vertices, degree and depth, respectively. For some \(v \in U_\tau\), let \(\Lambda(v)\) be the set of \(v\)'s children, \(\Gamma(v)\) be the set of \(v\)'s offspring leaves and \(\Delta_v\) be the length of edges from \(v\) to its children.

\textbf{Definition 3.2} \((k\text{-sibling-linked hierarchical well-separated tree})\). Let \(\tau\) be a \(k\text{-HST}\). We associate each \(v \in U_\tau\) a metric \(d_v : \Lambda_v \times \Lambda_v \to \mathbb{R}\) satisfying

\[ \forall u, w \in \Lambda(v), d_v(u, w) \leq 2\Delta_v \quad (2) \]
For some \( v \in U \), and \( u, w \in \Lambda(v) \), a sibling link \((u, w)\) is an edge of length \( d_m(u, w)\) connecting \( u \) and \( w \). We call the graph obtained by combining \( \tau \) and all the sibling links a \( k\)-sibling linked hierarchical well-separated tree (\( k\)-SLHST).

For a \( k\)-SLHST \( T \), we use \( d_T \) to denote its shortest path metric. The notions such as \( \text{dep}(\tau), \text{deg}(\tau), \Delta_v, \Lambda_v \) and \( \Gamma_v \) are naturally extended to \( k\)-SLHSTs. We say a \( k\)-SLHST \( T \) supports \( X \) if \( X \) is the leaves of \( T \).

To avoid confusion, for a \( k\)-SLHST \( T \) that supports \( X \), we use \( p, q \) to denote points in \( X \), as well as the leaves of \( T \), and \( u, v \) to denote inner vertices in a \( k\)-SLHST, if possible.

### 3.1 Embedding \( X \) into a distribution of \( k\)-SLHSTs

Now, we are going to describe our algorithm for embedding metric \((X, d_X)\) to a distribution of dominating \( k\)-SLHSTs. The algorithm is almost the same as the HST embedding algorithm in [11], except that we use \( n^{1/\beta} \) as the scale-decreasing factor, instead of 2. The embedding is pretty bad (it will incur a distortion of \( O(n^{\epsilon}) \)). As we’ll show, after we insert the sibling-links to the HST, the distortion becomes very small. We partition a set of radius \( \tau \) to clusters of radius \( r \) in \( \log(\tau/\beta) \) steps and in each step we reduce the radius by a factor of 2.

Despite its similarity with the tree embedding algorithm of [11], we give our algorithm here, for the integrality of the paper. Algorithm 1 partitions a metric \( Y \) in to sets of smaller radius, and it will be used in by algorithm 2.

#### Algorithm 1 partition\((Y, d_Y, r)\)

**Input:** A metric \((Y, d_Y)\) and a scale \( r \), \( r \) is guaranteed to be at least \( 8 \text{log} R \)/2.

**Output:** A partition of \( Y \) into clusters of radius smaller than \( r \) : \( \{Y_i : i \in [s]\} \)

1. Select a \( r \)-2-net \( S \subset Y \);
2. Randomly choose a permutation \( \pi \) for \( S \) and a number \( \beta \in [1, 2) \);
3. for \( i = 1, 2, \cdots, |S| \) do
   4. \( Y_i \leftarrow \text{Ball}_Y(\pi(i), \beta r/2) \setminus \bigcup_{j=1}^{i-1} Y_j \);
5. end for
6. return \( \{Y_i : i \in [|S|], Y_i \neq \emptyset\} \)

**Claim 3.3.** The returned \( \{Y_i : i \in [s]\} \) from algorithm 1 is indeed a partition, i.e., \( \forall i \neq j, Y_i \cap Y_j = \emptyset \) and \( \bigcup_{i \in [s]} Y_i = Y \).

**Algorithm 2 embedding_into_SLHST\((X, d_X, \epsilon)\)**

**Input:** A metric \((X, d_X)\) and \( 0 < \epsilon \leq 1/3 \) such that \( 1/\epsilon \) is an integer.

**Output:** A \( k\)-SLHST \( T \):

1. \( h \leftarrow \text{log} R_X \);
2. \( C \leftarrow \{X\}, C' \leftarrow \emptyset, V_r = \{v_X\}, E_r \leftarrow \emptyset \);
3. for \( i = 1, \cdots, h \) do
   4. for Every \( Y \in C \) do
      5. \( C'' \leftarrow \text{partition}(Y, d_X, \epsilon 2^{h-i}) \);
   6. for Every \( Y' \in C'' \) do
      7. \( V_r \leftarrow V_r + \{Y'\}, E_r \leftarrow E_r + \{(v_Y, v_{Y'})\} \);
   8. end for
   9. \( C' \leftarrow C' + C'' \);
   10. end for
   11. \( C \leftarrow C', C' \leftarrow \emptyset \);
   12. end for
13. \( a \leftarrow \lceil \epsilon \log n/\alpha X \rceil, b \leftarrow \text{random integer from } [a]; \)
14. \( L = \{i : 0 \leq i \leq h, i \equiv b(\text{mod } a)\} + \{0, h\} \);
15. Shrink \( \tau \) at level set \( L \): we remove all level-\( i \) vertices for all \( i \notin L \), and directly connect level-\( j \) vertices to their level-\( i \) ancestors using edges of length \( 2^{h-i} \), for \( i < j \) and \( i, j \) adjacent numbers in \( L \); let \( \tau' \) be the new tree.
16. Let \( i < j \) be two adjacent numbers in \( L \), for a level-\( i \) vertex \( v_Y \) and two level-\( j \) vertices \( v_{Y'}, v_{Y''} \) whose parent in \( \tau' \) is \( v_Y \) (so, \( Y', Y'' \subset Y \) ), \( d_{v_Y} = d_X(c_{Y'}, c_{Y''}) \), where \( c_{Y'} \) is the center of \( Y' \) (i.e., the radius of \( Y' \) is the maximum distance from \( c_{Y'} \) to some other point in \( Y' \)).
17. return the \( k\)-SLHST \( T \) defined by \( \tau' \) and \( d_{v_Y} \).

To avoid confusion, we use “rank” instead of “level” to denote the positions of vertices in \( \tau \), \( \tau' \) and \( T \). We say a vertex \( v \in v_x \) has “rank” \( i \), if the path from \( v \) to the root in \( \tau \) contains \( i \) edges. The rank of a vertex in \( T \) (or \( \tau' \)) is just the rank of its correspondent vertex in \( \tau \). Notice that if some vertex \( v \in U_T = U_{\tau'} \) has rank \( i \), then \( i \in L \).

**Lemma 3.5.** The algorithm 2 actually returns a
SLHST, i.e., the associated metrics satisfy equation \[3\]. Furthermore, \( \deg(T) \leq n^{O(c)} \).

Proof. Let \( v_Y \in U_T \) be a rank \( i \) vertex and \( v_{Y'}, v_Y' \) be two of its children in \( T \). Notice that \( v_Y, v_{Y'} \) has the same rank \( j \), where \( i \) and \( j \) are adjacent in \( L \). Then \( d_{v_Y}(v_{Y'}, v_{Y''}) = d_X(c_Y, c_{Y''}) \leq 2R_{v_Y} \leq 2^{h-1+i} = 2\Delta_{v_Y} \).

The degree of \( \tau \) is \( 2^{O(n)} \), by claim \( \[3\] \). The shrinking operation collapses \( a \) levels into 1 level, and thus, the degree becomes at most \( 2^{O(n)a} = 2^{O(\alpha/\epsilon \log n/\alpha)} = n^{O(c)} \).

**Lemma 3.6.** The random SLHST \( T \) that algorithm \[3\] returns supports \( X \) and satisfies:

1. \( \forall p, q \in X, d_T(p, q) \geq d_X(p, q) \);
2. \( \forall p, q \in X, \mathbb{E}(d_T(p, q)) \leq O(\alpha/\epsilon) d_X(p, q) \).

Proof. Fix two points \( p \) and \( q \). Let \( v_Y, v_Y' \) be the two highest-rank vertices in the shortest path between \( p \) and \( q \) in \( T \), and \( v \) be their parent. Let the rank of \( v_Y \) and \( v_Y' \) be \( i \). If \( i = h \), clearly \( d_T(p, q) = d_X(p, q) \); if \( i < h \),

\[
\frac{d_T(p, q)}{d_X(p, q)} \geq d_X(v_Y, v_Y') = d_X(c_Y, c_{Y'}) + 2^{h-i+1} \\
\geq d_X(p, q) - d_X(p, c_Y) - d_X(q, c_Y') + 2^{h-i+1} \\
\geq d_X(p, q) - 2^{h-i} - 2^{h-i} + 2^{h-i+1} \\
= d_X(p, q);
\]

So, we’ve proved the first property.

Let \( P_i \) be the probability that \( p \) and \( q \) are separated at rank \( i \) in \( \tau \). Obviously, \( \sum_{i=1}^{h} P_i = 1 \), and from \[11\], we have

\[
\sum_{i=1}^{h} P_i 2^{h-i} \leq O(\log n) d_X(p, q)
\]

Now, we fix a tree \( \tau \). Suppose \( p \) and \( q \) are separated at rank \( i \) in \( \tau \). Let \( c_Y \) and \( c_Y' \) be the rank \( j \) ancestors of \( p \) and \( q \), respectively. If \( i \leq h - a + 1 \), \( p \) and \( q \) may be separated at rank \( i, i+1, \ldots, i+a-1 \) in \( \tau' \), each with probability \( 1/a \). The expected \( d_T(p, q) \), over all possible \( b_i \) with \( D_i \), is at most

\[
\frac{1}{a} \sum_{j=i}^{i+a-1} \left( d_X(c_Y, c_Y') + 2^{h-1} 2^{h-j'} \right) \\
\leq \frac{1}{a} \sum_{j=i}^{i+a-1} \left( d_X(p, q) + 2^{h-j} + 2^{h-j} + 4 \times 2^{h-j} \right) \\
\leq d_X(p, q) + \frac{12}{a} 2^{h-i}
\]

If \( p \) and \( q \) are separated at rank \( i \) in \( \tau \), for some \( i > h - a + 1 \), \( D_i \) is at most

\[
\frac{1}{a} \left( \sum_{j=i}^{h} d_X(c_Y, c_Y') + (i + a - 1 - h) d_X(p, q) \right) \\
\leq d_X(p, q) + \frac{1}{a} \sum_{j=i}^{h} \left( 2^{h-j} + 2^{h-j} + 2 \sum_{j'=j}^{h-1} 2^{h-j'} \right) \\
\leq d_X(p, q) + \frac{12}{a} 2^{h-i}
\]

The last inequality comes from the fact that \( a = \lfloor \epsilon \log n/\alpha \rfloor \).

**3.2 Decomposition of EMD over a SLHST**

Before showing the main lemma of this subsection, we introduce some notions.

Let \( T \) be a SLHST that supports \( X \). For a distribution \( \mu \in \mathcal{P}_X \) and a vertex \( v \in U_T \), define \( \hat{\mu}_v = \sum_{t \in \tau(v)} \mu_t \) and \( \hat{\nu}_v = \bigoplus_{u \in \Lambda(v)} \mu_u \). Define extended EMD (or EEMD) between \( \hat{\mu}_v \) and \( \hat{\nu}_v \) to be

\[
EEMD(\hat{\mu}_v, \hat{\nu}_v) = \min_{\pi : \Lambda_v \times \Lambda_v \to \mathbb{R}} \ EEMD^\pi(\hat{\mu}_v, \hat{\nu}_v)
\]

\[
EEMD^\pi(\hat{\mu}_v, \hat{\nu}_v) = \sum_{u, w \in \Lambda_v} \pi(u, w) d_v(u, w)
\]

\[
+ \Delta_v \left( \sum_{u \in \Lambda_v} \mu_u - \sum_{u, w \in \Lambda_v} \pi_{u, w} \right)
\]

\[
+ \Delta_v \left( \sum_{u \in \Lambda_v} \nu_u - \sum_{u, w \in \Lambda_v} \pi_{u, w} \right)
\]

where \( \pi \) ranges over all transportation functions sat-
Lemma 3.7. \[
EMD_T(\mu, \nu) = \sum_{v \in U_T} EEMD_v(\hat{\mu}_v, \hat{\nu}_v)
\]

Proof sketch. We can view distributions \( \mu \) and \( \nu \) as supplies and demands on the leaves of the tree. The allowed operation is moving \( \epsilon \) amount of supplies (or demands) along some edge \((u, v)\), which costs \( \ell(u, v) \). We can cancel out the same amount of supplies and demands on the same vertex for free. We can further restrict the moving direction, so that the moves can no longer go downwards.

We show that the best strategy is to match supplies and demands locally, i.e., it can not move supplies along \((u, v)\) while moving demands along \((w, v)\), for some \( v \in U_T \) and \( u, w \in \Lambda_v \). Otherwise we would moving some amount of supplies directly from \( u \) to \( w \) and then match the same amount of demands at \( w \). This will make the cost smaller, since \( d_\epsilon(u, w) < \Delta_u + \Delta_v \). Thus, in the best matching, each vertex \( v \) will receive \( \max(\mu_v - \nu_v, 0) \) amount of supplies and \( \max(\nu_v - \mu_v, 0) \) amount of demands and cancels out all but \(|\mu_v - \nu_v| \) amount of supplies or demands, which will be sent to its parent. This is exactly the sum of \( EEMD_v \)s over all \( v \in U_T \).

Lemma 3.8. If \( X \) can be embedded into a set \( \mathcal{Y} \) of dominating metrics, with distortion \( \beta \), then, \( EMD_X \) can be embedded into \( \{ EMD_Y : Y \in \mathcal{Y} \} \), with distortion \( \beta \).

Proof. It’s easy to see that \( EMD_Y \) dominates \( EMD_X \). Let \( \pi : X \times X \to \mathbb{R} \) be the best transportation function for \( EMD_X(\mu, \nu) \), and \( \psi : Y \times Y \to \mathbb{R} \) be the best transportation function for \( EMD_Y(\mu, \nu) \). By the definition of EMD, we have
\[
EMD_X(\mu, \nu) = \sum_{p,q \in X} \pi(p, q)d_X(p, q)
\]
\[
EMD_Y(\mu, \nu) = \sum_{p,q \in X} \psi_Y(p, q)d_Y(p, q)
\]

Thus,
\[
\begin{align*}
\mathbb{E}_{Y \in \mathcal{Y}} EMD_Y(\mu, \nu) &= \mathbb{E}_{Y \in \mathcal{Y}} \sum_{p,q \in X} \psi_Y(p, q)d_Y(p, q) \\
&\leq \mathbb{E}_{Y \in \mathcal{Y}} \sum_{p,q \in X} \pi(p, q)d_Y(p, q) \\
&= \sum_{p,q \in X} \pi(p, q) \mathbb{E}_{Y \in \mathcal{Y}} d_Y(p, q) \\
&\leq \beta \sum_{p,q \in X} \pi(p, q) d_X(p, q) \\
&\leq \beta \mathbb{E} EMD_X(\mu, \nu)
\end{align*}
\]

Lemma 3.9. If \( T \) be the distribution of dominating SLHSTs that \( O(\alpha_X/\epsilon) \)-approximate \( X \), then \( EMD_X(\mu, \nu) \leq \mathbb{E}_{T \sim T} \left( \sum_{v \in U_T} EEMD_{\Lambda_v}(\hat{\mu}_v, \hat{\nu}_v) \right) \leq O(\alpha_X/\epsilon) EMD_X(\mu, \nu) \)

Proof. This lemma immediately follows lemma 3.7 and lemma 3.8.

4 Importance sampling, proof of theorem 1.1

After we decomposed \( EMD_X \) into \( \sum_{v \in U_T} EEMD_v \), we’ll use importance sampling to choose a element \( v \in U_T \), as mentioned in section 2. The probability that \( v \) is selected should roughly be the ratio of \( EEMD_v \) and \( EMD_T \). The following lemma from [14] tells us how many samples are needed.

Lemma 4.1 ([14]). Let \( Z_1, \ldots, Z_s \geq 0, Z = \sum_i Z_i \) and \( \epsilon = Z_i/Z \). Let \( p_i, p_{i+1}, \ldots, p_s \) be some numbers \( p_i \geq \epsilon \), and \( \sum_{i=1}^s p_i = 1 \), for some \( \gamma \). Consider a random variable \( S \) such that \( Pr[S = Z_i/p_i] = p_i \); note that \( E[S] = Z \). Then, \( \Pr \left[ \sum_{i \in [M]} S_i - Z > 0.5Z \right] \leq e^{-\Omega(M/\gamma)} \), where \( S_1, S_2, \ldots, S_M \) are independent copies of \( S \).

Proof. With probability 1,
\[
0 \leq S \leq \max_{i \in [s]} \frac{Z_i}{p_i} = \max_{i \in [s]} \frac{Zq_i}{p_i} \leq \gamma Z
\]

By Chernoff bound, we have
\[
\Pr \left[ \sum_{i \in [M]} S_i - Z > 0.5Z \right] \leq e^{-\Omega(M/\gamma)}.
\]
To apply lemma 4.1, we need a way to approximate each $EEMD_v$ within a reasonable ratio. Embedding the metric $d_v$ into distribution of trees can give an estimation that is always at least $EEMD_v$, and at most a logarithmic factor times $EEMD_v$ in expectation (see [8] and [11]). However, to allow a good estimation, the number of trees in the support should be $\Omega(|\Lambda_v|)$. Then, we need roughly $\sum_{v \in U_T} |\Lambda_v|^2 = O(n \max_i |\Lambda_v|)$ running time for the importance sampling, as opposed to $O(n)$ stated in the theorem. We can do slightly better by using the min of the EMDs, instead of the average.

**Lemma 4.2.** Let $(Y, d_Y)$ be a metric space such that $|Y| \leq n$, and $T$ be a distribution of dominating trees that approximate $(Y, d_Y)$ up to $O(\log n)$ factor, i.e., $\mathbb{E}_{t \in T} d_t(p, q) \leq O(\log n) d_Y(p, q)$. If we randomly choose $s = O(\log n)$ trees $\tau_1, \tau_2, \cdots, \tau_s$ from $T$, then for every pair of distributions $\mu, \nu$ over $Y$, 

$$\min_{i \in [s]} EMD_{\tau_i}(\mu, \nu) \leq O(\log n) EMD_Y(\mu, \nu)$$

with probability at least $1 - 1/n$.

**Proof.** For 1 tree $\tau$, with probability 1/2, $EMD_{\tau}(\mu, \nu) \leq O(\log n) EMD_Y(\mu, \nu)$. So, with probability at least 1 - $1/n$, there exists a $i \in [s]$ satisfying $EMD_{\tau_i}(\mu, \nu) \leq O(\log n) EMD_Y(\mu, \nu)$.

Now we can proceed to prove theorem 1.4.

**Proof sketch of theorem 1.4.** In the preprocessing, we run algorithm 2 to construct a SLHST $T_v$ with $\epsilon$ to be decided shortly. For each metric $(\Lambda_v, d_v)$ associated with an inner vertex $v \in U_T$, we choose $s = O(\log n)$ dominating trees $\tau_{v,1}, \tau_{v,2}, \cdots, \tau_{v,s}$, from the distribution of dominating trees that $O(\log n)$-approximates $d_v$. The data structure includes $T_v$, all $\tau_{v,s}$ and all sub-metrics of $X$ restricted to $\Lambda_v$ for $v \in U_T$.

The time to sample a SLHST is $O(n^2)$ and the time to compute all $\tau_{v,s}$ is still $O(n^2)$, implying the preprocessing time is $O(n^2)$.

The size of the data structure is $\tilde{O}(\sum_{v \in U_T} |\Lambda_v|^2) = n^{1+\Theta(\epsilon)}$. The $|\Lambda_v|^2$ space is for small metrics $d_v$, $s$ and sub-metrics of $X$ restricted to $\Lambda_v$.

Let $\mu, \nu \in \mathcal{P}_X$ be the two distributions in the query. With probability at least 0.9, the following two things happen:

1. $EMD_T = \sum_{v \in U_T} EEMD_v$ approximates $EMD_X$ with approximation ratio $O(\alpha/\epsilon)$.

2. For every $v \in U_T$, $\min_{i \in [s]} EEMD_{\tau_v}(\hat{\mu}_v, \hat{\nu}_v) \leq O(\log n) EEMD_v(\hat{\mu}_v, \hat{\nu}_v)$, by lemma 3.3 and lemma 4.2.

Assuming the above two things happen, we can do the importance sampling, using $\min_{i \in [s]} EEMD_{\tau_v}(\hat{\mu}_v, \hat{\nu}_v)$ as the weight for $v$. The probability that $v$ is selected in the importance sampling is at least $\frac{1}{O(\log n) EMD_T}$. So, by lemma 4.1, we only need $O(\log n)$ samples to approximate $EMD_T(\mu, \nu)$ up to a constant factor. This total time for importance sampling is $O(n)$, since $\sum_{v \in U_T} |\Lambda_v| = O(n)$.

Let $v$ be a vertex selected by the importance sampling. We compute $EEMD_v(\hat{\mu}_v, \hat{\nu}_v)$ by using standard Hungarian algorithm. $|\Lambda_v|$ is at most $n^\Theta(\epsilon)$, for small enough $\epsilon$, computing $EEMD_v$ takes $O(n)$ time.

In all, with preprocessing time $O(n^2)$, data structure of size $O(n^{1+\epsilon})$, we can approximate $EMD_X(\mu, \nu)$ up to a $O(\alpha_X/\epsilon)$ factor in $O(n)$ time. Notice that we can remove the $O(.)$ notion in the exponent, by losing a constant factor in the approximation ratio.

**5 Binary importance sampling, proofs of theorem 1.2, 1.3 and 1.4.**

In this section, we describe an encoding scheme, where each distribution $\mu$ is encoded into a linear code $F(\mu)$ and $EMD_X(\mu, \nu)$ can be approximated using sub-linear time, when $F(\mu)$ and $F(\nu)$ are given.

Recall that, in the algorithm for theorem 1.4, we need to do the importance sampling, where we estimate $EEMD_v$ for every $v \in T$. To design a sub-linear estimation time algorithm, we must avoid estimating $EEMD_v$ for every $v$.

This can be done by using a binary sampling method where, we maintain a set $S$ of possible outputs, initially $U_T$ and during each iteration, set $S$ is divided into 2 equal subsets, and replaced with one of the subset with probability proportional to the weight of the subset. The process repeats $O(\log n)$ times, until there’s only 1 element left in $S$, which is the output of the importance sampling.

The above method requires a good estimation for the total weight of a subset. In section 4, we used the min of $l_1$ norms as the estimation for the weight of an element. Then the total weight of a subset is the sum of mins, which seems hard to estimate. We’ll use a different estimation : the sum of $l_1$ norms,
which is still a \( l_1 \) norm. The total weight of a subset is again the sum of \( l_1 \) norms, equivalent to a \( l_1 \) norm. There are good sketching schemes for \( l_1 \) metric, for example, [17] used a random linear map \( g : l_1 \rightarrow \mathbb{R}^k \) as the sketching of a \( l_1 \) vector, such that the \( L_1 \) norm of \( x \) is approximated by the median of \(|g_1(x)|, |g_2(x)|, \ldots, |g_k(x)|\), where \( g_i \) is the \( i \)-th coordinate of \( g \). In the sketch function, each \( g_i \) is a linear function of \( x \), where the linear coefficients are chosen from the Cauchy distribution of scale parameter 1, i.e. \( g_i = \sum \alpha_j x_j, \alpha_j \sim C(0,1) \). The value \( k \) determines how well the norm is approximated, as in lemma 5.2

Before giving the lemma, we first define:

**Definition 5.1** (\( \rho \)-good). For some \( 0 < \rho < 0.1 \), we say the sketch function \( g \) is \( \rho \)-good for a fixed \( x \), if

\[
(1 - \rho) |x|_1 \leq \text{median}(|g_1(x)|, |g_2(x)|, \ldots, |g_k(x)|) \leq (1 + \rho) |x|_1
\]

where \( g_1, g_2, \ldots, g_k \) are \( k \) coordinates of \( g \).

**Lemma 5.2.** Let \( g : l_1 \rightarrow \mathbb{R}^k \) be the \( l_1 \) sketch function, with \( k = c/\rho^2 \) for some \( 0 < \rho < 0.1 \) and integer \( c \). Then, for a fixed \( x \in l_1 \), \( g \) is \( \rho \)-good with probability at least \( 1 - e^{-\Omega(c)} \).

We leave the proof of lemma 5.2 to the appendix. For a set of \( N \) elements, we fix a binary sampling tree, where each node is a subset of \([N]\). A set is equal to the union of its two child sets of half the size. The root of the tree is \([N]\), and the \( N \) leaves are \([\{1\}, \{2\}, \ldots, \{N\}]\). Let \( S_N \) be the family of all subsets of \([N]\) in this binary sampling tree; notice that \(|S_N| = O(N)\), and \( \sum_{S \in S_N} |S| = O(N) \). Our encoding \( f : (l_1)^N \rightarrow \mathbb{R}^{\Omega(Nk)} \) is defined as

\[
f(x_1, x_2, \ldots, x_N) = \bigoplus_{S \in S_N} g \left( \bigoplus_{i \in S} x_i \right).
\]

**Algorithm 3** \( b_{\text{import}}(\text{sample}(f(x_1, x_2, \ldots, x_N))) \)

**Input:** Encoding \( f \) for \( N \) vectors \( x_1, x_2, \ldots, x_N \).

**Output:** A pair \( (t, P) \), where \( t \in [N] \) and \( P_t \) is the probability that the algorithm selects \( t \);

1. \( S \leftarrow [N], P \leftarrow 1; \)
2. while \( |S| > 0 \) do
3. Let \( S_1 \) and \( S_2 \) be the two children of \( S \) in the binary sampling tree;
4. Extract \( \hat{g}_1 = g \left( \bigoplus_{j \in S_1} x_j \right) \) and \( \hat{g}_2 = g \left( \bigoplus_{j \in S_2} x_j \right) \) from the encoding \( f \);
5. \( W_1 \leftarrow \text{median} \left( |\hat{g}_1|, |\hat{g}_2|, \ldots, |\hat{g}_k| \right) \) and \( W_2 \leftarrow \text{median} \left( |\hat{g}_1|, |\hat{g}_2|, \ldots, |\hat{g}_k| \right) \);
6. Let \( i \) be 1 with probability \( W_1/W_1 + W_2 \) and 2 with probability \( W_2/W_1 + W_2 \);
7. \( S \leftarrow S_i, P \leftarrow P \times W_i/W_1 + W_2 \);
8. end while
9. return \( (t, P) \) where \( t \) is the unique element in \( S \);

**Proof.** If all occurrences of \( g \) in \( f \) are \( 1/(10 \log N) \)-good, the median of absolute values of the sketch will always give the \( l_1 \) norm of the pre-image, up to a factor of \( 1 \pm 1/(10 \log N) \). Thus the probability that algorithm 3 selects \( t \) is at least

\[
(1 - \frac{1}{10 \log N}) \log N \frac{|x_t|_1}{1 + \frac{1}{10 \log N}} > 0.5 \frac{|x_t|_1}{1 + \sum_{i=1}^N |x_i|_1},
\]

where the exponent \( \log N \) comes from the depth of the binary sampling tree. By definition 5.4 \( f \) is successful.

Then we apply algorithm 3 to compute the EMD over a SLHST metric. As shown in lemma 5.7 the EMD over a SLHST \( T \) is the sum of \( EEMD_{i, S} \) over all \( v \in U_T \). Each underlying metric \( d_v \) can be embedded into distribution of dominating trees, with distortion logarithmic in the size of the metric. In order to make the preprocessing efficient, we need a small set of dominating trees. [7] gives exactly what we want:

**Theorem 5.6** (7). Given a metric \( (X, d_X) \) with \(|X| = n\), there is a set of \( O(n \log n) \) dominating trees and a probability on them, such that the metric \( d_X \) is approximated by the set of dominating tree metrics, with distortion \( O(\log n \log \log n) \). Moreover, there is a polynomial algorithm which gives the set and the probability.

Then, by lemma 5.8 (or 8), the EEMD over \( (v, d_v) \) can be embedded into the average EEMD
of $O(|\Delta_v| \log |\Delta_v|)$ trees, which is equivalent to the norm of a $|\Delta_v|^2 \log |\Delta_v|$ dimensional $L_1$, with distortion $O(\log |\Delta_v| \log |\Delta_v|)$. We use $h_v : \mathbb{R}^{N(v)} \to \mathbb{R}^{O(|\Delta(v)|)}$ to denote the linear function whose $l_1$ norm approximates $E E M D_v$.

For a SHST $T$, let’s list all the vertices in $U_T = v_1, v_2, \ldots, v_N$. Define :

$$F_T(\mu) = f(h_{v_1}(\hat{\mu}_{v_1}), \ldots, h_{v_N}(\hat{\mu}_{v_N})) \oplus \bigoplus_{i \in [N]} \hat{\mu}_{v_i} \quad (4)$$

Then, algorithm 4 shows how to approximate the EMD in sub linear time, if the encodings are given.

Algorithm 4 $\text{approx}_\text{E M D}(T, F_T(\mu), F_T(\nu))$

1: $\text{Sum} \leftarrow 0$;
2: Extract $f_\mu, f_\nu = f(h_{v_1}(\hat{\mu}_{v_1}), \ldots, h_{v_N}(\hat{\mu}_{v_N}))$ from $F_T(\mu)$, and $f_\mu, f_\nu = f(h_{v_1}(\hat{\nu}_{v_1}), \ldots, h_{v_N}(\hat{\nu}_{v_N}))$ from $F_T(\nu)$;
3: for $i \leftarrow 1, 2, \ldots, \lceil \log^2 n \rceil$ do
4: \quad $(t, P) = \text{import_sample}(f_\mu - f_\nu)$;
5: Extract $\hat{\mu}_i$ from $F_T(\mu)$ and $\hat{\nu}_i$ from $F_T(\nu)$;
6: $\text{Sum} \leftarrow \text{Sum} + E E M D_{\nu_i}(\hat{\mu}_i, \hat{\nu}_i)/P$;
7: end for
8: return $\text{Sum}/[\log^2 n]$.

Lemma 5.7. If all occurrences of $f$ in $F_T(\mu - \nu) = F_T(\mu) - F_T(\nu)$ are successful, algorithm 4 outputs a number between $0.5E E M D_T(\mu, \nu)$ and $1.5E E M D_T(\mu, \nu)$ with probability at least $0.9$.

Proof. If all occurrence of $f$ in $F_T(\mu - \nu)$ are successful, algorithm 4 chooses $t$ with probability at least

$$0.5\hat{h}_t(\hat{\mu}_t, \hat{\nu}_t) \geq \frac{0.5E E M D_T(\hat{\mu}_t, \hat{\nu}_t)}{O(\log n \log \log n) \sum_{i \in [N]} E E M D_i(\hat{\mu}_i, \hat{\nu}_i)} = \frac{1}{O(\log n \log \log n)} E E M D_T(\mu, \nu)$$

By lemma 4.1, the probability that the algorithm outputs a number between $0.5E E M D_T(\mu, \nu)$ and $1.5E E M D_T(\mu, \nu)$ is at least

$$1 - e^{-\Omega(\log^2 n / \log n \log \log n)} \geq 0.9.$$

Proof sketch of theorem 1.2, 1.3 and 1.4. We first prove theorem 1.2 and then show how this it implies theorem 1.3 and 1.4.

The preprocessing stage is almost the same as the algorithm for theorem 1.1 except that we choose $s = O(|\Delta_v| \log |\Delta_v|)$ dominating trees $\tau_{e,1}, \tau_{e,2}, \ldots, \tau_{e,s}$ for $v$ using the algorithm in [7]. The preprocessing time is $O(n^2)$, and the size of the data structure is $O(n^{1+\epsilon})$.

We can suppose we have $E E M D_T$ approximates $E E M D_X$ with $O(\alpha_X/\epsilon)$ distortion, because this happens with high probability.

Then we compute the encoding $F_T(\mu)$ and $F_T(\nu)$ using formulas 3 and 4. The time to compute $f$ is $k$ times the dimension of the input, which is $\sum_{v \in U_T} O(|\Delta(v)|) = O(n^{1+\epsilon})$ in $F_T(\mu)$. The second part of $F_T(\mu)$ can be ignored. The time to compute the encoding is $O(kn^{1+\epsilon})$. The size of the encoding is $O(Nk) = O(nk)$.

If we set $k$ to $O(\log^3 n)$, then by lemma 5.7, every occurrence of $g$ is $O(1/\log n)$-good with probability $1 - O(1/n^3)$. Totally, there are $O(|U_T|) = O(n)$ occurrences of $g$ in $F_T(\mu)$, thus, with probability $1 - O(1/n^3)$, every occurrence of $g$ is $O(1/\log n)$-good, which implies, by lemma 5.6, $f(h_{v_1}(\hat{\mu}_{v_1}), \ldots, h_{v_N}(\hat{\mu}_{v_N}))$ is successful. This finally implies algorithm will output a number between $0.5 E E M D_T(\mu, \nu)$ and $1.5 E E M D_T(\mu, \nu)$ with probability at least $0.9$, by lemma 5.7. The number $O(\alpha_X/\epsilon)$-approximates $E E M D_X(\mu, \nu)$.

Since $k = O(\log^3 n)$, the time need to compute an encoding is $O(kn^{1+\epsilon}) = O(n^{1+\epsilon})$, and the size of an encoding is $O(nk) = O(n)$. The binary importance sampling takes time $O(1)$, the bottleneck of algorithm 4 is computing the $E E M D_v$s, which takes total time $\tilde{O}(n^2)$. In algorithm 4, we don’t really compute $F_T(\mu - \nu)$. As the encoding is linear, whenever we want to read a number from $F_T(\mu - \nu)$, we read the numbers in $F_T(\mu)$ and $F_T(\nu)$ and then do the subtraction.

Theorem 1.2 immediately implies theorem 1.3. Alice computes $F(\mu)$ and Bob computes $F(\nu)$, then, they communicate to simulate algorithm 4. The communication complexity is $O(n^2)$, with approximation ratio $\alpha_X/\epsilon$.

For theorem 1.4, we can store the encodings for all the $s$ distributions in the preprocessing stage and when a query comes, run algorithm 4. We can run the preprocessing stage $O(\log s)$ times independently and store $O(\log s)$ data structures so that with high probability, the EMD between every pair is reserved.

6 Conclusions

We have demonstrated a almost linear algorithm for estimating EMD over doubling metrics up to a constant factor. We also derived an encoding-based sub-
linear time constant approximation algorithm, which may have further applications.

An interesting direction to pursue would be to find a sketching scheme for EMD over doubling metrics. [4]'s sketching scheme benefits from the fact that all the small grids are the same. While in our case, the small metrics are different. If we can embed the EMD over the small metrics to any uniform normed space up to a constant factor, we can use the technique in [4] to derive a sketch scheme.

References

[1] On lipschitz embedding of finite metric spaces in hilbert space. Journal Israel Journal of Mathematics, pages 46–52, 1985.

[2] Pankaj Agarwal and Kasturi Varadarajan. A near-linear constant-factor approximation for euclidean bipartite matching? In SCG '04: Proceedings of the twentieth annual symposium on Computational geometry, pages 247–252, New York, NY, USA, 2004. ACM.

[3] Pankaj K. Agarwal, Alon Efrat, and Micha Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. In SCG '95: Proceedings of the eleventh annual symposium on Computational geometry, pages 39–50, New York, NY, USA, 1995. ACM.

[4] Alexandr Andoni, Khanh Do Ba, Piotr Indyk, and David Woodruff. Efficient sketches for earthmover distance, with applications. 2009.

[5] Aaron Archer, Jittat Fakcharoenphol, Chris Harrelson, Robert Krauthgamer, Kumal Talwar, and Eva Tardos. Approximate classification via earthmover metrics. In SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms, pages 1079–1087, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[6] Ya'ir Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In In 37th Annual Symposium on Foundations of Computer Science, pages 184–193, 1996.

[7] Moses Charikar, Chandra Chekuri, Ashish Goel, Sudipto Guha, and Serge Plotkin. Approximating a finite metric by a small number of tree metrics. In FOCS '98: Proceedings of the 39th Annual Symposium on Foundations of Computer Science, page 379, Washington, DC, USA, 1998. IEEE Computer Society.

[8] Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In STOC '02: Proceedings of the thirty-fourth annual ACM symposium on Theory of computing, pages 380–388, New York, NY, USA, 2002. ACM.

[9] Scott Cohen and Leonidas Guibas. The earth mover’s distance: Lower bounds and invariance under translation. Technical report, Stanford, CA, USA, 1997.

[10] Scott Cohen and Leonidas Guibas. The earth mover’s distance under transformation sets. In ICCV ’99: Proceedings of the International Conference on Computer Vision-Volume 2, page 1076, Washington, DC, USA, 1999. IEEE Computer Society.

[11] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In STOC ‘03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, pages 448–455, New York, NY, USA, 2003. ACM.

[12] H. N. Gabow and R. E. Tarjan. Faster scaling algorithms for network problems. SIAM J. Comput., 18(5):1013–1036, 1989.

[13] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In 44th Annual IEEE Symposium on Foundations of Computer Science, pages 534–543, October 2003.

[14] Piotr Indyk. A near linear time constant factor approximation for euclidean bichromatic matching (cost). In SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pages 39–42, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[15] Piotr Indyk and Nitin Thaper. Fast color image retrieval via embeddings. Technical report, 2003.

[16] R. Krauthgamer, J. R. Lee, M. Mendel, and A. Naor. Measured descent: A new embedding method for finite metrics. In 45th Annual IEEE Symposium on Foundations of Computer Science, pages 434–443. IEEE, October 2004.

[17] Eyal Kushilevitz, Rafail Ostrovsky, and Yuval Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces, 1998.
A Proof of lemma 5.2

Proof. For a fixed $x$ and $i \in [k]$, the distribution of $g_i(x)$ is $\mathcal{C}(0, |x|_1)$. Let $\gamma = |x|_1$, and

$$f(t) = \begin{cases} 
0 & \text{if } t < 0 \\
\frac{2}{\pi} \frac{\gamma}{t+\gamma} & \text{if } t \geq 0
\end{cases}$$

be the probability density function $|g_i|_s$. $F$ is the correspondent cumulative function:

$$F(t) = \begin{cases} 
0 & \text{if } t < 0 \\
\frac{2}{\pi} \arctan \left( \frac{t}{\gamma} \right) & \text{if } t \geq 0
\end{cases}$$

For some $0 < \delta < 0.1$, define $Y_i = 1$ if $g_i < F^{-1}(1/2 - \delta)$ or $g_i > F^{-1}(1/2 + \delta)$ and 0 otherwise. Using Chernoff bound, we have

$$\Pr \left[ \sum_{i=1}^{k} Y_i \geq k/2 \right] \leq e^{-\delta^2 k/8}$$

If $k = c/\delta^2$, the above probability is at most $e^{-\Omega(c)}$. For small enough $\delta$, $F^{-1}(1/2 - \delta) = (1 - \Theta(\delta))\gamma$ and $F^{-1}(1/2 + \delta) = (1 + \Theta(\delta))\gamma$. $\sum_{i=1}^{k} Y_i < n/2$ implies the absolute median falls between $(1 - O(\delta))\gamma$ and $(1 + O(\delta))\gamma$. □