Approximating High-Dimensional Earth Mover's Distance as Fast as Closest Pair

Anonymous Authors

Abstract

We give a reduction from $(1 + \varepsilon)$-approximate Earth Mover's Distance (EMD) to $(1 + \varepsilon)$-approximate Closest Pair (CP). As a consequence, we improve the fastest known approximation algorithm for high-dimensional EMD. Here, given $p \in [1, 2]$ and two sets of n points $X, Y \subset (\mathbb{R}^d, \ell_p)$, their EMD is the minimum cost of a perfect matching between X and Y, where the cost of matching two vectors is their ℓ_p distance. Further, CP is the basic problem of finding a pair of points realizing $\min_{x \in X, y \in Y} ||x - y||_p$. Our contribution is twofold:

- We show that if a $(1+\varepsilon)$-approximate CP can be computed in time $n^{2-\phi}$, then a $1 + O(\varepsilon)$ approximation to EMD can be computed in time $n^{2-\Omega(\phi)}$.

- Plugging in the fastest known algorithm for CP [ACW16b], we obtain a $(1+\varepsilon)$-approximation algorithm for EMD running in time $n^{2-\tilde{\Omega}(\varepsilon^{1/3})}$ for high-dimensional point sets, which improves over the prior fastest running time of $n^{2-\Omega(\varepsilon^2)}$ [AZ23].

Our main technical contribution is a sublinear implementation of the Multiplicative Weights Update framework for EMD. Specifically, we demonstrate that the updates can be executed without ever explicitly computing or storing the weights; instead, we exploit the underlying geometric structure to perform the updates implicitly.

1 Introduction

In this paper, we study algorithms for the Earth Mover's Distance (EMD) which is a fundamental measure of dissimilarity between two sets of point in a metric space [PC19; San15; Vil+08] (also known as the Wasserstein-1 metric). Specifically, we consider the ground metric $(\mathbb{R}^d, \ell_p)$, for $p \in [1, 2]$, and given two subsets of n vectors, $X = \{x_1, \ldots, x_n\}, Y = \{y_1, \ldots, y_n\} \subset \mathbb{R}^d$, the Earth Mover's distance (EMD) between X and Y is

$$\mathsf{EMD}(X, Y) = \min_{\substack{\pi : [s] \to [s] \\ \text{bijection}}} \sum_{i=1}^{n} \|x_i - y_{\pi(i)}\|_p$$

In other words, $\mathsf{EMD}(x, y)$ is the value of a min-cost flow in the complete bipartite geometric graph: the n points of x become vertices on the left-hand side and have uniform supply, the n points of y become vertices on the right-hand side and have uniform demand, and every edge between x_i and y_j appears with cost $\|x_i - y_j\|_p$ and infinite capacity.

A natural algorithmic approach to compute EMD is to first construct the geometric graph and then utilize the best graph algorithm for min-cost flow. However, this graph has n^2 edges, resulting in an at least quadratic time algorithm. On the other hand, the input has size $O(nd)$, thus it is feasible that faster algorithms exist. In fact, there is a extensive line-of-work on approximation algorithms for EMD running in sub-quadratic time; to avoid constructing the complete graph, algorithms must exploit the geometric structure of $(\mathbb{R}^d, \ell_p)$ [Aga+17; Aga+22; AIK08a; And+14; AS14b; AZ23; BR24; Cha+23; Cha02; FL23; IT03; KNP19; Roh19; SA20].

The Spanner Approach. A standard approach to avoid constructing the full geometric graph is to employ *geometric spanners*, which are sparse graphs where the shortest path distances approximates the geometric distances. Since the runtime of min-cost flow graph algorithms scale with the number of edges [ASZ20; Che+22a; Li20a; She17], sparser spanners will yield faster approximations for EMD.[1] In low-dimensional space, where $d = o(\log n)$, one can construct $(1 + \varepsilon)$ approximate spanners with $n \cdot O(\varepsilon^{-d})$ edges [Alt+93; Cla87; RS91]. However, until recently it was not known whether sub-quadratic $(1 + \varepsilon)$ approximate spanners exist for high-dimensional spaces for $\varepsilon < 2.4$.[2] This was addressed by [AZ23], who designed such spanners (crucially using additional Steiner vertices) with $n^{2-\Omega(\varepsilon^3)}$ undirected edges or $n^{2-\Omega(\varepsilon^2)}$ directed edges. The main application of their work is a $(1+\varepsilon)$-approximation algorithm for EMD over $(\mathbb{R}^d, \ell_p)$ running in time $n^{2-\Omega(\varepsilon^2)}$. To date, this is the only sub-quadratic $(1 + \varepsilon)$ approximation of high-dimensional EMD.

The fact that the runtimes of all state-of-the-art EMD approximations are directly tied to the best available spanner construction motivates our central question: can this dependency be circumvented? Specifically:

> *Do there exists $(1 + \varepsilon)$-approximation algorithms to high-dimensional EMD with running times faster than that of the best spanner constructions?*

We answer the above question affirmatively, and thereby improve the runtime of best known $(1+\varepsilon)$ approximation for EMD. Our key technical contribution enabling this is a black-box reduction

---

[1]Using the almost-linear time algorithm for capacitated min-cost flow [Che+22a], it suffices to construct a *directed* spanner (i.e., a directed graph whose shortest paths approximate pairwise distances). If using near-linear, approximate, uncapacitated min-cost flow [ASZ20; Che+22a; Li20a; She17], the spanner must be undirected.

[2]For large constant approximations, it is known that C-approximate $n^{1+O(1/C)}$ sized spanners exist [HIS13].

from EMD to the approximate closest pair (CP) problem. Here, the CP problems is: given two sets $X, Y \subseteq \mathbb{R}^d$, return a pair $(x, y) \in X \times Y$ such that $\|x - y\|_1 \leqslant (1 + \varepsilon) \min_{x' \in X, y' \in Y} \|x' - y'\|_1$.[3] Closest pair is a cornerstone problem in computational geometry [ACW16b; Val15], and is intimately related to the nearest neighbor search problem [AI08; And09; HIM12; IM98]. Crucially, CP is amenable to powerful algorithmic techniques outside of spanners, most notably the polynomial method [ACW16b]. These techniques have yielded CP algorithms with runtimes faster than the best known spanner constructions. Specifically, we prove the following:

Theorem 1 (Main Theorem, informal version of Theorem 5). *If there exists an algorithm for $(1 + \varepsilon)$-approximate closest pair on $(\mathbb{R}^d, \ell_1)$ with running time $T_{CP}(n, \varepsilon) = n^{2 - \phi(\varepsilon)}$, then there exists an algorithm that computes a $(1 + O(\varepsilon))$-approximation to $\mathsf{EMD}(A, B)$ over $(\mathbb{R}^d, \ell_p)$ for any $p \in [1, 2]$ in time $n^{2 - \Omega(\phi(\varepsilon))}$.*

Theorem 1 provides a fine-grained reduction, translating runtime improvements for CP directly into runtime improvements for EMD, losing only a constant in ε and $\phi(\varepsilon)$. The existence of such an efficient reduction is perhaps surprising. Namely, while exact closest pair admits a simple $O(n^2)$ algorithm, computing EMD, which is a global matching optimization task, appears significantly more complex. Indeed, a natural greedy approach for EMD—repeatedly finding the closest available pair (a, b), matching them, and removing them—is known to yield very poor approximations [RT81]. Despite the failure of this simple reduction, Theorem 1 demonstrates that, via a considerably more involved reduction, approximate EMD is essentially no harder than approximate CP.

By plugging in the fastest known algorithm for approximate CP in high-dimensional ℓ_1 by Alman, Chan, and Williams [ACW16b], which runs in time $n^{2 - \tilde{\Omega}(\varepsilon^{1/3})}$, we obtain:

Corollary 1.1. *There exists an algorithm that, with high probability, computes a $(1 + \varepsilon)$-approximation to $\mathsf{EMD}(A, B)$ between two n-point sets $A, B \subset (\mathbb{R}^d, \ell_1)$ in time $dn + n^{2 - \Omega(\varepsilon^{1/3} / \log(1/\varepsilon))}$.*

This is a significant improvement over the $n^{2 - O(\varepsilon^2)}$-time spanner-based algorithm of [AZ23], reducing the exponent of ε by a factor of 6. Moreover, we note that it is perhaps unlikely that the complexity of Corollary 1.1 can be matched by the Spanners approach. Specifically, there is a non-trivial barrier to improving the sparsity of $(1 + \varepsilon)$ high-dimensional spanners beyond $n^{2 - \varepsilon}$; specifically, all such spanner constructions are crucially based on Locality Sensitive Hashing (LSH) [AZ23; HIS13; Ind01], and therefore only as efficient as the best algorithm for all-pairs nearest neighbor search via LSH, for which the best known constructions require $n^{2 - O(\varepsilon)}$ time. Thus, improving the spanners approach beyond $n^{2 - O(\varepsilon)}$ would likely require a major breakthrough in LSH algorithms.

1.1 High-Level Overview and Conceptual Contribution.

We now provide a high-level overview of our algorithm, and elaborate on additional details in the Technical Overview (Section 2). Our approach to Theorem 1 is to solve the standard transshipment linear program (Figure 1). However, a direct approach faces an immediate obstacle: the primal linear program has $O(n^2)$ variables, making it impossible to even represent a feasible solution within our target subquadratic runtime. While the dual program offers a potential advantage, as it has only $O(n)$ variables (allowing representation of a dual solution), it presents its own challenge: checking

---

[3]For both EMD and CP, we focus on the ℓ_1 norm. Via standard reductions [JS82], one can embed ℓ_p for $p \in [1, 2]$ into ℓ_1 with $(1 + \varepsilon)$ error, making ℓ_1 the more general space.

dual feasibility requires evaluating $O(n^2)$ constraints, which again is too expensive. Despite this strong restriction, we nonetheless demonstrate how we can implement the multiplicative weights update (MWU) framework to solve the dual in time sublinear in the number of constraints.

We emphasize that while many works have used the MWU framework to solve linear programs, including for EMD [Li20a; Zuz23], these works typically assume the ability to explicitly store and manipulate the MWU weights. To our knowledge, our work is the first to tackle the setting where, due to the strict subquadratic time requirement, the weights cannot be stored explicitly. Instead, our algorithm must continuously operate on the implicit geometric representation of the input points. This necessitates a careful integration of geometric techniques and optimization principles.

Specifically, in Section 6 we show that the MWU update step can be reduced to the task of generating $\tilde{O}(n)$ samples from a class of distributions λ defined over over pairs $[n] \times [n]$. The probability $\lambda_{i,j}$ is proportional to $\exp\left(K_{i,j} \cdot \frac{1}{\|x_i - y_j\|_1}\right)$, where $K_{i,j}$ is a value depends on the current dual variables. Again, we emphasize that we cannot *explicitly* represent this distribution due to its size. Instead, we must rely solely on the implicit geometric representation of the points. Our key insights for sampling efficiently from λ are as follows:

1. **Reducing to fixed K:** We first demonstrate that we can partition $X \times Y$ into combinatorial rectangles $\{X_1 \times Y_1, \ldots, X_r \times Y_r\}$ so that $K_{i,j}$ is (roughly) the same for all (i, j) in a rectangle. By sampling from each rectangle $X_i \times Y_j$ independently, this allows us to reduce to the case where $K_{i,j} = K$ is fixed for all pairs i, j (Lemma 7.9).

2. **Ideal Sampling with Distance Gaps:** Hypothetically, suppose that every distance was of the form $(1 + \varepsilon)^t$ for an integer t, creating discrete gaps. Since K is fixed (from step 1), $\lambda_{i,j} \propto \exp\left(K/\|x_i - y_j\|_1\right)$, so we could use an $(1 + \varepsilon)$-approximate CP algorithm to iteratively find the pairs with smallest distances (thus highest probabilities $\lambda_{i,j}$). The CP algorithm can stop after it has reached a level t such that there are at least $n^{1+\phi}$ pairs at distance $(1 + \varepsilon)^{t+1}$. After explicitly storing the pairs with distance at most $(1 + \varepsilon)^t$, we can use rejection sampling for the remaining pairs. Rejection sampling will be efficient ($\tilde{O}(n^{1-\phi})$ time per sample) because the distance gaps guarantee a $1/n^{1-\phi}$ fraction of remaining points all share the *next* highest probability level.

3. **Challenge: Real Distances & Approximate CP:** Note that simply rounding the distances in MWU to the nearest power of $(1 + \varepsilon)$ fails. The core issue is that the approximate CP algorithm works on the original geometry of the points, not these conceptual rounded distances. Namely, a $(1 + \varepsilon)$ approximate CP algorithm may return pairs with (rounded) distance $(1 + \varepsilon)^{t+1}$ before it finds all pairs with (rounded) distance $(1 + \varepsilon)^t$. This is a critical issue since $\lambda_{i,j}$ depends exponentially on the distance (and K can be polylog(n)); thus, missing even one closer pair can unacceptably skew the sampling distribution.

4. **Flawed Attempt: CP-Adaptive Rounding:** To address this issue, we could try rounding distances adaptively based on the CP output within one iteration (e.g., round down if the pair is found by CP, round up otherwise). Thus, undiscovered points have lower probabilities by construction. This approach has a fundamental flaw: the rectangles $X_k \times Y_k$ which are input to the CP algorithm will change after each MWU iteration; in particular, in one iteration the CP algorithm may find (i, j) and round it down, and on the next iteration the CP algorithm may not find the pair. Thus, we must decide on a fixed rounding scheme up front.

3

5. **Solution: Fixed Randomized Rounding:** Our key technique is showing the success of a randomized rounding strategy. Specifically, we sample a set $S \subset X \times Y$ of size $n^{2-\phi/2}$; we then round distances down for pairs $(i,j) \in S$, and round distances up otherwise.

   - **Why it works:** For any fixed set L of $n^{1+\phi}$ pairs, the intersection $|S \cap L|$ will be sufficiently large (at least $n^{1+\phi/2}$) with high probability. If L is the set of pairs with distance between $(1+\varepsilon)^t$ and $(1+\varepsilon)^{t+1}$, then by rounding down the distances in $S \cap L$ we ensure there's a sufficiently large pool of points with the "maximal" probability among those not explicitly found by CP, making rejection sampling efficient (needed in step 2).
   - **Addressing Critical Dependencies:** This seems promising, except for a potentially fatal fault: S influences λ, which influences the samples from λ, which influence the dual variables and thus the rectangles $X_k \times Y_k$ that are constructed in the next iteration. Thus, the samples S are not independent of the sets L (as defined by the pairs at a given distance inside of a rectangle) we need them to intersect with. However, the crucial observation is that the rectangles depend *only* on the $\tilde{O}(n)$ samples from λ on the previous step, not all n^2 probabilities affected by S. Since there are only $T = \mathrm{polylog}(n)$ iterations, the number of possible sample histories (and thus possible rectangle partitions) is at most $2^{T \cdot \tilde{O}(n)}$. We can thus union bound over all these possibilities to show that our single, fixed random sample S works with high probability for any set L arising within any rectangle that the algorithm actually encounters (details in Section 8).

2 Technical Overview

Earlier in Section 1.1 we gave a bird-eye view of our algorithmic approach. In this section, we sketch the techniques used to prove Theorem 1 in further detail. We first devise a template algorithm that, if implemented naively, runs in quadratic time. Our main technical contribution will be to show how this template algorithm can be implemented via a small number calls to a closest pair algorithm. We will then exploit the existence of fast algorithms for approximate closest pair [ACW16b].

Fix two input sets $X = \{x_1 \ldots x_n\}, Y = \{y_1 \ldots y_n\} \subseteq ([1, \Phi]^d, \ell_1)$. We extend the definition EMD to arbitrary supply vectors and denote with $\mathrm{EMD}(\mu, \nu)$ the cost of minimum uncapacited flow routing the supply $\mu_1 \ldots \mu_n \in [0,1]$ located at $x_1 \ldots x_n$ to demands $\nu_1 \ldots \nu_n \in [0,1]$ located at $y_1 \ldots y_n$. In particular, the EMD between X and Y can be denoted by $\mathrm{EMD}(\mathbf{1}, \mathbf{1})$, or simply EMD.

In Section 2.1, we will describe the template algorithm for approximating EMD, based on solving the dual LP via multiplicative weights update (MWU). The key step to implementing the update step in the MWU procedure is to sample from the distributions λ described inSection 1.1. Then, in Section 2.2, we describe our scheme to sample from this class of distributions in subquadratic time.

2.1 Template Algorithm for Approximating EMD

Our starting point for approximating EMD will be to solve the standard linear program for the bipartite min-cost flow problem, shown below in Figure 1.

Figure 1: Linear program for EMD.

Our goal will be to approximate the optimal objective value to this LP in subquadratic time. However, notice that the primal formulation has n^2 variables, in the form of the flows $\{\boldsymbol{\gamma}_{ij}\}_{i,j\in[n]}$. Thus, we cannot even write down a solution to the primal, let alone directly implement iterative update methods. On the other hand, the dual has only $2n$ variables $\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i\}_{i\in[n]}$. Thus, in principle it could possible to compute a dual solution $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ in subquadratic time, since its size is only $O(n)$. Of course, the challenge is now that the dual has n^2 constraints, thus *verifying* whether a given solution $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is feasible is likely impossible in subquadratic time. However, in what follows we will show that we can approximately verify feasibility efficiently.

For now, we first we describe an "inefficient" algorithm that computes a $(1 + \varepsilon)$-approximation of EMD in time $\tilde{O}(n^2)$. Let $C_{ij} := ||x_i - y_j||$. For $t > 0$, consider the following polytope:

$$\Gamma_t = \left\{ (\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{R}^n \times \mathbb{R}^n : \frac{1}{t} \cdot \left(\sum_{i=1}^{n} \boldsymbol{\alpha}_i - \sum_{j=1}^{n} \boldsymbol{\beta}_j \right) - \max_{i,j} \frac{|\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j|}{C_{ij}} > 0 \right\}.$$

By duality, it is straightforward to prove that EMD $> t$ if and only if $\Gamma_t \neq \varnothing$. By performing an search over t in powers of $(1 + \varepsilon)$, the problem of approximating EMD can be solved by leveraging a procedure that, if $\Gamma_{(1+\varepsilon)t} \neq \varnothing$, returns a certificate $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \Gamma_t$. To this end, we employ the framework of multiplicative weights update (MWU), described in the following.

Multiplicative Weights Update (MWU). Observe that the polytope Γ_t is defined by $2n^2$ constraints (each absolute value constraint corresponds to two linear constraints), which we index with triplets $(i, j, \sigma) \in [n] \times [n] \times \{\pm 1\}$. We define a set of weights $w_{i,j,\sigma}$, one for each constraints, and initialize all of them to 1. This induces a distribution over constraints λ such that $\lambda_{i,j,\sigma}$ is proportional to $w_{i,j,\sigma}$. Importantly, again, we note that we cannot explicitly maintain the distribution λ, as it is a $O(n^2)$ sized object. Nevertheless, in what follows we show that it will suffice to sample from λ to compute the MWU update.

The classic MWU procedure operates in rounds: at each round, we define a "special" constraint (1) by taking a λ-weighted combination of the constraints

$$\frac{1}{t} \cdot \left(\sum_{i=1}^{n} \boldsymbol{\alpha}_i - \sum_{j=1}^{n} \boldsymbol{\beta}_j \right) - \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \lambda_{i,j,\sigma} \frac{\sigma(\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j)}{C_{ij}} > 0, \tag{1}$$

5

then, we design an oracle CERTIFY(λ) that should return $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ such that:

Invariant (I): $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ satisfies the special constraint in Equation (1).

Invariant (II): $\left| \frac{1}{t} \left(\sum_{\ell=1}^{n} \boldsymbol{\alpha}_\ell - \sum_{\ell=1}^{n} \boldsymbol{\beta}_\ell \right) \right|, \left| \frac{\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j}{C_{ij}} \right| \leqslant \text{polylog}(n)$ for all $i, j \in [n]$.

Then, the weights are updated to penalize constraints that are violated by $(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

$$w_{i,j,\sigma} \leftarrow w_{i,j,\sigma} \cdot \exp\left(\eta \cdot \left(\frac{1}{t} \cdot \left(\sum_{i=1}^{n} \boldsymbol{\alpha}_i - \sum_{j=1}^{n} \boldsymbol{\beta}_j \right) - \frac{\sigma \cdot (\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j)}{C_{ij}} \right) \right) \quad \text{for constant } \eta > 0. \quad (2)$$

After $\mathsf{R} = \text{polylog}(n)$ rounds, we show that if the two invariants are always satisfied by the output of the CERTIFY(λ) procedure, then the average $(\bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{\beta}})$ of the dual variables $(\boldsymbol{\alpha}^{(1)}, \boldsymbol{\beta}^{(1)}), \ldots,$ $(\boldsymbol{\alpha}^{(\mathsf{R})}, \boldsymbol{\beta}^{(\mathsf{R})})$ returned at each round satisfies the constraints in $\Gamma_{(1-2\varepsilon)t}$ (Lemma 6.8). We are left to implement the oracle CERTIFY(λ).

Probabilistic tree embedding. Before we implement CERTIFY(λ), we first recall a fundamental tool in high-dimensional geometry: the probabilistic tree embedding. Specifically, this technology allows us to embed points in $(\mathbb{R}^d, \ell_1)$ into a tree metric with polylogarithmic *expected* distortion [AIK08b]. Concretely, given the set of points $X \cup Y \subseteq ([1, \Phi]^d, \ell_1)$, there exists a distribution over trees $\mathcal{T}$ satisfying 1–5 below:

1. Each leaf of $\mathcal{T}$ corresponds to an element of $X \cup Y$.

2. $\mathcal{T}$ has depth $\log \Phi$.

3. Every edge connecting a node at level ℓ with one at level $\ell + 1$ has weight $\frac{\Phi}{2^\ell}$.

4. $||x_i - y_j|| \leqslant \Theta(\log n) \cdot \mathsf{d}_\mathcal{T}(x_i, y_j)$, with probability $1 - 1/\text{poly}(n)$ (where $\mathsf{d}_\mathcal{T}$ is the distance in the tree).

5. $\mathbf{E}[\mathsf{d}_\mathcal{T}(x_i, y_j)] \leqslant \Theta(\log \Phi) \cdot ||x_i - y_j||$.

We remark that probabilistic tree embeddings do *not* satisfy yield Bi-Lipschitz distortion guarantees. Namely, we *cannot* guarantee that with high probability:

$$\mathsf{d}_\mathcal{T}(x_i, y_j) \leqslant \Theta(\log \Phi) \cdot ||x_i - y_j|| \quad (3)$$

This deficit will become relevant shortly.

For the purposes of approximating EMD to a $\text{polylog}(n, \Phi)$ factor, however, it turns out that expected polylog distortion suffices. Specifically, it is not difficult to show that the EMD in the tree metric, $\text{EMD}_\mathcal{T}(\mu, \nu)$, enjoys the same distortion guarantees: $\text{EMD}_\mathcal{T}(\mu, \nu) \geqslant \Theta(1/\log n) \cdot \text{EMD}(\mu, \nu)$; $\mathbf{E}[\text{EMD}_\mathcal{T}(\mu, \nu)] \leqslant \Theta(\log \Phi) \cdot \text{EMD}(\mu, \nu)$. The advantage of this embedding is that now the EMD within the tree metric is relatively simple to compute, as it can be written exactly as:

$$\text{EMD}_\mathcal{T}(\mu, \nu) = \sum_{v \in \mathcal{T}} \frac{\Phi}{2^\ell} \cdot \left| \sum_{x_i \in v} \mu_i - \sum_{y_i \in v} \nu_i \right|, \quad (4)$$

6

where $x \in v$ means that x belong to the subtree rooted at the internal node v. The formula (4) will be useful in our implementation of CERTIFY(λ) below. Specifically, we remark that while alterantive methods may exist for designing polylog "rough" approximations to EMD, the tree-embedding formulation, specifically (4), will be critical in allowing $\mathrm{EMD}_{\mathcal{T}}(\mu, \nu)$ to be estimated via sampling.

Certify(λ) implementation. We now describe an *inefficient* implementation of the CERTIFY(λ) procedure, which will run in quadratic time; we will show thereafter how this procedure can be approximated by a subquadratic sampling-based algorithm.

To construct feasible variables $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, we first define the following supply and demand vectors:

$$\mu_i = t \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \qquad \text{and} \qquad \nu_j = t \sum_{i=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}}.$$

Observe that $\gamma_{ij} = t(\lambda_{i,j,1} - \lambda_{i,j,-1})/C_{ij}$ is a feasible flow for $\mathsf{EMD}(\mu, \nu)$ with cost at most t. Thus, $\mathsf{EMD}(\mu, \nu) \leqslant t$. Moreover, if we are promised that $\Gamma_{(1+\varepsilon)t} \neq \varnothing$, then $\mathrm{EMD}(\mathbf{1}, \mathbf{1}) > (1 + \varepsilon)t$. Hence, using a "triangle inequality" for EMD:

$$\Theta(\log n) \cdot \mathrm{EMD}_{\mathcal{T}}(\mathbf{1} - \mu, \mathbf{1} - \nu) \geqslant \mathrm{EMD}(\mathbf{1} - \mu, \mathbf{1} - \nu) \geqslant \mathrm{EMD}(\mathbf{1}, \mathbf{1}) - \mathrm{EMD}(\mu, \nu) > \varepsilon \cdot t.$$

By Equation (4) applied to $(\mathbf{1} - \mu, \mathbf{1} - \nu)$, there exists a level $T_\ell \subseteq \mathcal{T}$ of the tree $\mathcal{T}$ such that

$$\sum_{v \in T_\ell} \frac{\Phi}{2^\ell} \left| |X_v| - |Y_v| - t \sum_{\substack{i \in [n] \\ x_i \in v}} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} + t \sum_{\substack{j \in [n] \\ y_j \in v}} \sum_{i=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \right| = \tilde{\Omega}(\varepsilon t). \quad (5)$$

Set $\boldsymbol{\alpha}_i^* = \boldsymbol{\beta}_j^* = \frac{\Phi}{2^\ell} \cdot s_v$ for $x_i, y_j \in v$, where $s_v \in \{-1, 1\}$ denote the sign which realizes the absolute value on the left-hand side of Equation (5). With some additional work, we can rearrange the terms in Equation (5) to obtain

$$\sum_{i=1}^{n} \boldsymbol{\alpha}_i^* - \sum_{j=1}^{n} \boldsymbol{\beta}_j^* - t \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \lambda_{i,j,\sigma} \cdot \sigma \cdot \frac{\boldsymbol{\alpha}_i^* - \boldsymbol{\beta}_j^*}{C_{ij}} = \tilde{\Omega}(\varepsilon t),$$

so $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ satisfies invariant (I).

Challenge: satisfying invariant (II) requires Bi-Lipschitz tree embeddings. Notice that by construction $\boldsymbol{\alpha}_i^* - \boldsymbol{\alpha}_j^* \neq 0$ only if x_i and y_j belong to different subtrees at level ℓ. If Property 3 held for the tree $\mathcal{T}$, then $\boldsymbol{\alpha}_i^* - \boldsymbol{\alpha}_j^* \neq 0$ would imply $C_{ij} = \tilde{\Omega}(\frac{\Phi}{2^\ell})$, and, in turn, $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ would satisfy invariant (II). Unfortunately, there exists a set of points that cannot be embedded into a tree satisfying both properties 4 and 3 for all pairs[4]. Thus, it will not be possible to achieve invariant (II) without a *Bi-Lipschitz* guarantee on the distortion of points. Moreover, satisfying invariant (II) is critical to the success of the MWU procedure.

---

[4]For example the set of n evenly spaced points on a circumference.

Obtaining Bi-Lipschitz guarantees via post-hoc perturbation. We address this challenge by designing a post-hoc perturbation $X' \cup Y'$ of the points in $X \cup Y$ such that:

- On $X' \cup Y'$, $\mathcal{T}$ satisfies properties 1–5, as well as the stronger property in Equation (3).

- For any fixed μ, ν, $\mathrm{EMD}_{X',Y'}(\mu, \nu) = (1 \pm \varepsilon) \cdot \mathrm{EMD}_{X,Y}(\mu, \nu)$.

Importantly, the perturbed points X', Y' *depends* on the realization of the random tree $\mathcal{T}$. Specifically, for each node $\boldsymbol{v} \in \mathcal{T}$ we draw a random $\boldsymbol{z_v} \in [0, \varepsilon \cdot \Phi/2^\ell]^{O(\log n)}$. For each point $\boldsymbol{x} \in X \cup Y$ we compute the root-to-leaf path $\boldsymbol{v}_1 \ldots \boldsymbol{v}_h = \boldsymbol{x}$ and add the vector $\sum_{i=1}^{h} \boldsymbol{z}_i$ to $\boldsymbol{x}$. Any two points whose least common ancestor is at level ℓ will have their distance increased by $\approx \frac{\Phi}{2^\ell}$. At the same time, the perturbation cannot make $\mathrm{EMD}_{X,Y}(\mu, \nu)$ much larger than $\mathrm{EMD}_{\mathcal{T}}(\mu, \nu)$, which, in turn, is at most $\Theta(\log \Phi) \cdot \mathrm{EMD}_{X,Y}(\mu, \nu)$.

2.2 Implementing the Template in Subquadratic Time

We now demonstrate how the above algorithmic template can be efficiently implemented. In particular, we first demonstrate how the the oracle $\mathrm{CERTIFY}(\lambda)$ can be implemented using $\tilde{O}(n)$ samples from λ, rather than requiring an explicit representation of all $2n^2$ probabilities. Secondly, we reduce the problem of sampling from λ to the problem of finding an approximate closest pair.

Implementing Certify(λ) with samples. First, we recall how the previously described $\mathrm{CERTIFY}(\lambda)$ procedure constructs $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$. Let

$$\mathcal{Q}_v(\lambda) := \frac{\Phi}{2^\ell} \cdot \left(|X_v| - |Y_v| - t \sum_{\substack{i \in [n] \\ x_i \in v}} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} + t \sum_{\substack{j \in [n] \\ y_j \in v}} \sum_{i=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \right) \quad (6)$$

First, we find a level $T_\ell \subseteq \mathcal{T}$ in the probabilistic tree satisfying Equation (5), which can be rewritten as $\sum_{v \in T_\ell} |\mathcal{Q}_v(\lambda)| = \tilde{\Omega}(\varepsilon t)$. Then, define $\boldsymbol{\alpha}_i^* = \boldsymbol{\beta}_j^* = \frac{\Phi}{2^\ell} \cdot \mathrm{sign}\left(\mathcal{Q}_v(\lambda)\right)$ for all $x_i, y_j \in v \in T_\ell$.

To implement $\mathrm{CERTIFY}(\lambda)$ with samples, we draw $s = \tilde{O}(n)$ i.i.d. samples $(i, j, \sigma) \sim \lambda$, define $\widehat{\lambda}$ as the empirical distribution of these samples, and run $\mathrm{CERTIFY}(\widehat{\lambda})$. We can verify that, if we have concentration of the form $\mathcal{Q}_v(\widehat{\lambda}) = \mathcal{Q}_v(\lambda) \pm O(\varepsilon \cdot t)$, then $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ satisfies invariants I and II. Thus, the correctness of $\mathrm{CERTIFY}(\widehat{\lambda})$ boils down to bounding the variance of $\mathcal{Q}_v(\widehat{\lambda})$ for one sample $(\boldsymbol{i}_S, \boldsymbol{j}_S, \boldsymbol{\sigma}_S) \sim \lambda$, which we do in the following. The stochastic part of $\mathcal{Q}_v(\widehat{\lambda})$ can be written as

$$t \cdot \sum_{\substack{i,j \in [n] \\ \sigma \in \{-1,1\}}} \sigma \cdot (\mathbf{1}\{x_i \in v\} - \mathbf{1}\{y_j \in v\}) \cdot \mathbf{1}\{(\boldsymbol{i}_S, \boldsymbol{j}_S, \boldsymbol{\sigma}_S) = (i, j, \sigma)\} \cdot \frac{\Phi/2^\ell}{C_{ij}} \quad (7)$$

If $x_i, y_j \in v$ (or $x_i \notin v, y_j \notin v$), the terms in the sum corresponding to i, j in Equation (7) cancel. Else, if $x_i \in u$ and $y_j \in v$ for two distinct nodes $u, v \in T_\ell$, then by property 3 of $\mathcal{T}$, we have

$$\frac{\Phi/2^\ell}{C_{ij}} = \frac{\Theta\left(\mathsf{d}_{\mathcal{T}}(x_i, y_j)\right)}{C_{ij}} = O(\log \Phi).$$

Each of the $2n^2$ summand in Equation (7) has variance at most $O(t^2 \cdot \log^2 \Phi)$. Thus, $s = \tilde{O}(n/\varepsilon^2)$ sample suffice to shrink the variance to $O(\varepsilon^2 \cdot t^2)$.

Interlude: sampling seems as hard as closest pair. In this interlude we suggest why closest pair seems the right tool for sampling from λ. Let $(\boldsymbol{\alpha}^{(1)}, \boldsymbol{\beta}^{(1)}) \ldots (\boldsymbol{\alpha}^{(r)}, \boldsymbol{\beta}^{(r)})$ be the dual solutions returned on the first r rounds of MWU. Let λ be the distribution proportional to the weights $w_{i,j,\sigma}$ updated r times as in Equation (2). Then,

$$\lambda_{i,j,\sigma} \propto \exp\left(\frac{\sigma \cdot \left(\sum_{\ell=1}^r \boldsymbol{\alpha}_i^{(\ell)} - \boldsymbol{\beta}_j^{(\ell)}\right)}{C_{ij}}\right). \tag{8}$$

According to invariant (II), we could have $|\boldsymbol{\alpha}_i^{(\ell)} - \boldsymbol{\beta}_j^{(\ell)}| \approx C_{ij} \cdot \mathrm{polylog}(n)$. Suppose that $C_{ij} = 1$ for all $i,j \in [n]$ besides $C_{i*,j*} = 1 - \varepsilon$ and that, for all $i,j \in [n]$, $\sum_{\ell=1}^r \boldsymbol{\alpha}_i^{(\ell)} - \boldsymbol{\beta}_j^{(\ell)} = K = \mathrm{polylog}(n)$. Then, $\lambda_{i,j,\sigma} \propto \exp(\sigma \cdot K/C_{ij})$ is overwhelmingly concentrated on $(i*, j*, +1)$. Thus, under the invariants proven so far, sampling from λ could be as hard as finding the closest pair. In the following, we prove that in fact sampling from λ can be reduced to closest pair.

2.3 Sampling from λ via Closest Pair

Roadmap. Sampling from λ as defined in Equation (8) presents multiple challenges, which we decouple as follows. To begin, suppose we instead were tasked with solving a restricted version of the problem:

<u>Restriction</u> (1) We assume that the term $\sum_{\ell=1}^r \boldsymbol{\alpha}_i^{(\ell)} - \boldsymbol{\beta}_j^{(\ell)}$ in Equation (8) is constant. Namely, we sample from $\lambda_{i,j,\sigma} \propto \exp(\sigma \cdot K/C_{ij})$.

<u>Restriction</u> (2) We assume that our closest pair algorithm returns all "close" pairs, where a pair is close if its distance is one of the smallest $O(n^{1+\phi})$ distances.

We will show that, with some effort, both restrictions can ultimately be lifted.

Sampling from an easier distribution. First, note that to sample from $\lambda_{i,j,\sigma} \propto \exp(\sigma \cdot K/C_{ij})$, it is sufficient to sample a variablle $\xi_{ij} \propto \exp(|K|/C_{ij})$ and thereafter apply a rejection sampling step to sample the sign $\sigma \in \{-1, 1\}$, which gives a constant runtime overhead. Interestingly, for very large K, ξ concentrates on the closest pair, thus sampling from ξ for arbitrary K is as hard as *exact*[5] CP, and the latter requires quadratic time [Wil18]. Intuitively, sampling form ξ requires to find a needle in a haystack. However, $(1+\varepsilon)$-approximate EMD is resilient to small perturbation of distances, which we can leverage to obviate this issue. Consider a *rounding* $\tilde{C}_{ij}$ of the distances in C_{ij} to the closest power of $1 + \varepsilon$, either up or down. Clearly, $\mathrm{EMD}_C \approx \mathrm{EMD}_{\tilde{C}}$, so, to approximate EMD, it suffices to fix a rounding $\tilde{C}_{ij}$ *upfront* and then run our whole algorithm with respect to $\tilde{C}_{ij}$. After rounding C_{ij}, in what follows we will see that a $(1 + \varepsilon)$-approximate CP algorithm is sufficient to sample from ξ.

Define the "t-prefix set" $\mathcal{L}_t$ as the set of pairs at distance at most $(1+\varepsilon)^t$. Now suppose we had a $(n^{2-\Omega(\phi)})$-time procedure `AllClosePairs`(t) that returns $\mathcal{L}_t$ as long as $|\mathcal{L}_{t+1}| = O(n^{1+\phi})$. By sampling uniformly random pairs $(i,j) \in [n] \times [n]$ and checking if $(i,j) \in \mathcal{L}_r$ (for every $r \geq 0$), we can easily find t such that: (i) $|\mathcal{L}_{t+1}| = O(n^{1+\phi})$ (ii) $\mathcal{L}_{t+2} \backslash \mathcal{L}_t$ is large, i.e., $|\mathcal{L}_{t+2 \backslash \mathcal{L}_t}| = \tilde{\Omega}(n^{1+\phi})$. We

---

[5]Even if we assume a realistic bound on K, e.g., $|K| = \mathrm{polylog}(n)$, sampling from ξ reduces to $(1/\mathrm{polylog}(n))$-approximate CP, that cannot be solved faster than $n^{2-o(1)}$ [Rub18].

then define the distances $\tilde{C}$ as a function of a set $S \subseteq [n] \times [n]$ (to be defined in the following), by rounding down the distances for all pairs (i,j) in S, and rounding up every other pair. So long as we have $|S| = n^{2-\Omega(\phi)}$ we can afford to compute the distances $\tilde{C}$ upfront and store them explicitly. Our sampling algorithm will partition the pairs $A \times B$ in two sets $\mathcal{E}$ and $\mathcal{I}$, and develop different procedures to sample from $\xi|_{\mathcal{E}}$ and $\xi|_{\mathcal{I}}$. Then, to sample from ξ we just need to estimate the total weight $\sum_{(i,j) \in A} \exp(|K|/\tilde{C}_{ij})$ for $A = \mathcal{E}, \mathcal{I}$.

First, we define $\mathcal{E}$ as the set of pairs handled explicitly $\mathcal{E} = \mathcal{L}_t \cup (S \cap \mathcal{L}_{t+1})$, and define $\mathcal{I}$ to be the set of pairs handled implicitly, i.e. $\mathcal{I} = [n]^2 \backslash \mathcal{E}$. Sampling from $\tilde{\xi}$ restricted to $\mathcal{E}$ can be done by explicitly computing individual weights $\exp(|K|/C_{ij})$. To sample from $\tilde{\xi}$ restricted to $\mathcal{I}$, we sample a uniform pair $(i,j) \in \mathcal{I}$ and then keep it with probability

$$\exp(K/\tilde{C}_{ij})/\exp(K/(1+\varepsilon)^{t+1}). \tag{9}$$

otherwise we reject it. Since all pairs $(i,j) \in \mathcal{I}$ satisfy $\tilde{C}_{ij} \geqslant (1+\varepsilon)^{t+1}$, the quantity (9) is a valid probability, and conditioned on not rejecting a pair it results in the correct distribution. Moreover, for at least $|S \cap \mathcal{L}_{t+2}| + |\mathcal{L}_{t+1} \backslash S|$ of these pairs, namely the points for which $\tilde{C}_{ij} = (1+\varepsilon)^{t+1}$, the quantity (9) is in fact equal to 1, and therefore these pairs are never rejected. Thus if we have

$$|S \cap \mathcal{L}_{t+2}| + |\mathcal{L}_{t+1} \backslash S| = n^{1+\Omega(\phi)}, \tag{10}$$

then the above rejection sampling method successfully generates a sample after $O(n^{1-\phi})$ attempts. Thus, we can generate the desired $\tilde{O}(n)$ samples from ξ using $n^{2-\Omega(\phi)}$, as is required too implement the CERTIFY(λ) oracle, in $n^{2-\Omega(\phi)}$ preprocessing time, plus $O(n^{1-\phi})$ time per sample. Thus, given that the set S satisfies the above constraints (and assuming the two restrictions stated earlier), this approach will successfully generate the necessary samples. This leads to teh following question.

Question 1. *Can we define S (and, thus, the rounding $\tilde{C}$) in advance so that it satisfies Equation (10) every time we call this subroutine?*

To understand our solution to Question 1, we must first delve into how we lift Restrictions (1) and (2).

Lifting Restriction (1): Reducing λ to $\propto \exp(\sigma \cdot K/C_{ij})$. We now show how to reduce the problem of sampling from λ as in Equation (8) to that of sampling from a distribution $\propto \exp(\sigma \cdot K/C_{ij})$.

Define $\bar{\beta} := \sum_{\ell=1}^r \alpha_\ell$, $\bar{\alpha} := \sum_{\ell=1}^r \beta\ell$. Notice that the coefficient $\bar{\alpha}_i - \bar{\beta}_j$ is highly structured. We can leverage this structure to reduce to the case of identical coefficients $\bar{\alpha}_i - \bar{\beta}_j = K$. First, we round $\bar{\alpha}_i - \bar{\beta}_j$ to the closest power of $(1+\chi)$, for some small[6] χ. Suppose that $\bar{\alpha}_i$ and $\bar{\beta}_i$ sorted in increasing order. Consider the set $Q_t \subseteq [n] \times [n]$ of pairs (i,j) such that $\bar{\alpha}_i - \bar{\beta}_j$ is rounded to $(1+\chi)^t$. Q_t also is highly structured: indeed its boundaries are described by *monotonic* sequences $(1,j_1) \ldots (n,j_n)$ where $j_1 \leqslant \cdots \leqslant j_n$. As a consequence[7], we can tile each Q_t with small combinatorial rectangles of size $s \times s$, with $s = \sqrt[4]{n}$, leaving only $n^{2-\Omega(1)}$ uncovered pairs in total.

On each combinatorial rectangle, sampling from λ corresponds to sampling from a distribution proportional to $\exp(\sigma \cdot K/C_{ij})$. As for the uncovered pairs, we can sample from them explicitly

---

[6]Taking $\chi = \frac{1}{\text{polylog}(n)}$ is sufficient to ensure that we sample from a distribution close enough in total variational (TV) distance to λ.

[7]Essentially, this is the same approximation argument used to show that monotonic functions are Riemann integrable.

since there are at most $n^{2-\Omega(1)}$ many of them. Then, it is sufficient to estimate the normalization constant (which can be done by sampling):

$$\sum_{(i,j,\sigma)\in\mathcal{C}\times\{\pm 1\}}\exp(\sigma\cdot K/C_{ij})$$

for each combinatorial rectangle $\mathcal{C}$. To sample, we first sample a combinatorial rectangle proportionally to its normalization constant, and then sample a pair within the sampled rectangle. If sampling from a single $s\times s$ combinatorial rectangle requires time $s^{2-\Omega(\phi)}$, then the overall running time of this reduction is $\frac{n^2}{s^2}\cdot s^{2-\Omega(\phi)}=n^{2-\Omega(\phi)}$.

Lifting Restriction (2): Retrieving all close pairs via CP. Note that fastest approximate CP algorithms [ACW16b; Val15] leverage similar algebraic methods to each other. Essentially, these methods lump together a batch $Y_i\subset Y$ of n^ϕ points in into individual vector v_i; then they show that a single dot product $\langle a,v_i\rangle$ is sufficient to detect if there exists $y\in Y_i$ such that $||x-y||$ is small. If that is the case, all distances in $\{a\}\times Y_i$ are computed. Interestingly, this method can be tweaked to return all close pairs, in the sense of Restriction (2). However, we would like a generic reduction to CP, and not rely on the specific structure of the aforementioned algorithms.

Question 2. *Can we transform any algorithm for approximate CP into an algorithm that return all (approximately) close pairs?*

We answer this question affirmatively. Specifically, we now show how to implement `AllClosePairs` in time $n^{2-\Omega(\phi)}$ by leveraging a $n^{2-\phi}$-time algorithm `CP` for $(1+\varepsilon)$-approximate CP. Up to rescaling ε, we can assume that $|\mathcal{L}_{t+3}|\ll n^{1+\phi}$. We subsample $X'\subseteq X$ and $Y'\subseteq Y$ with rate $1/z$ for $z:=\sqrt{n^{1+\phi}}$, then compute `CP`(X',Y'). A pair $(x,y)\in\mathcal{L}_t$ is guaranteed to be returned by `CP`(X',Y') as long as $(X'\times Y')\cap\mathcal{L}_{t+1}=\{(x,y)\}$. Condition on the event $x\in X'$ and $y\in Y'$, which happens with probability z^{-2}. Then, assuming $|\mathcal{L}_{t+1}|\leqslant|\mathcal{L}_{t+3}|\ll z^2$, a union bound over all pairs $(c,d)\in\mathcal{L}_{t+1}\cap(X\backslash\{x\})\times(Y\backslash\{y\})$ shows that, with constant probability, none of these pairs lies in $X'\times Y'$. Notice that the bound above is possible because the events $(c,d)\in X'\times Y'$ and $(x,y)\in X'\times Y'$ are independent. For pairs of the form (x,d) (likewise for (c,y)), the probability of $(x,d)\in X'\times Y'$ conditioned on $(x,y)\in X'\times Y'$ is $\deg_{\mathcal{L}_{t+1}}(x)/z$.

Therefore, we need to consider two cases:

1. If $\deg_{\mathcal{L}_{t+1}}(x)<z/3$, then the probability of any colliding pair $(c,d)\in\mathcal{L}_{t+1}\cap(X\times Y)\backslash\{(x,y)\}$ is at most $1-\Omega(1)$. Thus, the probability of `CP`(X',Y') returning (x,y) is $\Omega(z^{-2})$ and $\tilde{O}(z^2)$ repetitions suffice to find (x,y).

2. If $\deg_{\mathcal{L}_{t+1}}(x)\geqslant z/3$, then `CP`$(X',Y')$ returns x often, so, we shall see that can *detect* x using $\tilde{O}(z)$ repetitions and checking what fraction of the times a pair $(x,\cdot)$ is returned.

Either way, $\tilde{O}(z^2)=\tilde{O}(n^{1+\phi})$ calls to `CP`(X',Y') suffice. By standard concentration $|X'|,|Y'|\approx\frac{n}{z}$, thus `CP`$(X',Y')$ runs in time $(\frac{n}{z})^{2-\Omega(\phi)}$, which gives an overall running time of $n^{2-\Omega(\phi)}$.

To detect x in point 2, we need to ensure that, if $\deg_{\mathcal{L}_{t+1}}(x)\geqslant z/3$, then a pair $(x,\cdot)$ is returned often enough (i.e., with probability $\Omega(z^{-1})$). The probability that the $\mathcal{L}_{t+1}$-neighborhood of x intersects Y' is a large constant. On the other hand, the probability that any not-incident-to-x pair in $\mathcal{L}_{t+2}$ lies in $X'\times Y'$ is, by union bound over all $\mathcal{L}_{t+2}$ such pairs, small. Thus, with constant probability, the first event happens and the second event does not. These events are independent of

$x \in X'$, which happens with probability $1/z$. Thus, with probability $\Omega(z^{-1})$ both these events happen and x is the only point in X' incident to some edge in $\mathcal{L}_{t+2} \cap X' \times Y'$, so, $\mathtt{CP}(X', Y')$ returns $(x, \cdot)$.

Now that we have lifted both Restriction (1) and Restriction (2), all that remains is the answer Question 1. Namely, to define a set $S \subset [n] \times [n]$, so as to fix the rounding $\tilde{C}$, so that it satisfies Equation (10) every time we attempt to sample from a distribution λ.

Solving Question 1 via Fixed Randomized Rounding. The core challenge, posed in Question 1, is to define a rounding scheme $\tilde{C}$ (by specifying the set S of pairs to be rounded down) that guarantees Equation (10) holds every time the sampling scheme is called, so that the rejection sample step is efficient. Recall that Equation (10) requires that for the relevant distance levels $\mathcal{L}_{t+1}, \mathcal{L}_{t+2}$, the set S has a sufficiently large intersection with $\mathcal{L}_{t+2}$. Further recall that t is defined as the level for which $\mathcal{L}_{t+2}$ is "large" and $\mathcal{L}_{t+1}$ is not too "large". As discussed earlier in Section 1.1, the specific sets $\mathcal{L}_{t+1}, \mathcal{L}_{t+2}$ depend on the subproblem (i.e., the combinatorial rectangle $X_k \times Y_k$) being considered, which changes throughout the Multiplicative Weights Update (MWU) iterations. Importantly, these subproblems are a function of the output of the MWU algorithm on the last iteration, which depend on the prior distribution λ and therefore the rounding. Thus the sets $\mathcal{L}_{t+1}, \mathcal{L}_{t+2}$ are not independent of the rounding we choose.

Our solution, introduced conceptually in Section 1.1 and detailed technically in Section 8, is to use a randomized rounding strategy despite this dependency. We sample a single set $S \subseteq X \times Y$ of size $n^{2-\phi/2}$ uniformly at random at the beginning of the algorithm. Since $\mathcal{L}_{t+2}$ is defined to have size at least $n^{1+\phi}$, we will have $|S \cap \mathcal{L}_{t+2}| = n^{1+\Omega(\phi)}$ with probability $1 - \exp(-\Omega(|S|))$ if S were independent from $\mathcal{L}_{t+2}$. Unfortunately, as described above, it is not independent. Our approach to resolve this independence issue is to demonstrate that the combinatorial rectangles on a given step of MWU can be taken to be a deterministic function of the $\tilde{O}(n)$ samples drawn from the distribution λ on the last step. Thus, there are at most $2^{\tilde{O}(n)}$ possible rectangles (and thus possible sets $\mathcal{L}_r$) on each step. Since we only need to run MWU for $\mathrm{polylog}(n)$ iterations, we can union bound over the set of all $2^{\tilde{O}(n)}$ possible sets $\mathcal{L}_r$ that could ever occur as the relevant set in a given subproblem, thereby guaranteeing correctness.

Comparison with Sherman's Framework. In an influential paper, Sherman [She13] presented a framework for efficiently approximating flow problems with high accuracy. Sherman's preconditioning framework is very versatile and has thus been refined to obtain faster parallel algorithm for shortest path [Li20b] (see also the excellent introduction in Li's thesis [Li21]). In essence, Sherman's framework reduces $(1+\varepsilon)$-approximate uncapacitated minimum cost flow to $\mathrm{polylog}(n)$-approximate ℓ_1-*oblivious routing*.

Employed off-the-shelf, Sherman's frameworks falls short of providing improvements for $(1+\varepsilon)$-approximate EMD beyond the already discussed approach of building a spanner and solving the corresponding graph problem on the spanner. Importantly, current implementations of Sherman's framework (e.g., [Li20b]) do not appear to lend themselves to sublinear-time implementations.

To counter this, we design an ad-hoc oblivious embedding (Section 5) and give an alternate implementation of Sherman's framework (Section 6), which we show is amenable of sublinear-time implementation in the size of the LP (Section 7). The bulk of our work is to show that such sublinear implementation can be achieved by leveraging the geometry of the space.

3 Related Work

Other Approaches to Approximating EMD.   This focus of this paper is on "high-quality" $(1 + \varepsilon)$ approximations to high-dimensional EMD, where we achieve the best known runtime (Corollary 1.1), and for which ε is taken to be smaller than some constant. For this regime, there were no known subquadrtic algorithms prior to [AZ23], which we improve on here. For C-approximations, where C is a large constant, a classic result in metric geometry is that any n-point metric space admits a spanner of size $O(n^{1+1/C})$ (which, moreover, can be constructed efficiently in Euclidean space) [HIM12; PS89]. Via fast min-cost flow solvers [Che+22a], this yields a C-approximation to EMD in time $n^{1+O(1/C)+o(1)}$. Additionally, it is known that EMD over $(\mathbb{R}^d, \ell_1)$ can be embedded into a tree metric with $O(\log n)$ distortion in $\tilde{O}(n)$ time [Che+22b], resulting in an algorithm for EMD with the same runtime and approximation. We note that EMD cannot be computed exactly in high-dimensions in truly-subquadratic time (conditioned on the Orthogonal Vectors Conjecture or SETH), [Roh19], hence the emphasize on approximations.

Note that for low-dimensional spaces, where d is taken to be a constant, it is known that EMD can be approximated to $(1 + \varepsilon)$ in near-linear time [Aga+22; SA12], although incurring exponential in dimension factors (i.e. $(1/\varepsilon)^d$).

The closest pair problem itself is central to computational geometry. It is tightly related to nearest neighbor search [AI08; And09; HIM12; IM98], and indeed computing approximate nearest neighbors (ANN) for all points is sufficient to solve approximate CP. Given the best $(1 + \varepsilon)$ approximate nearest neighbor search algorithms run in time $O(n^{1+\varepsilon})$ [AR15], this yields a CP algorithm with the same runtime. In a suprising breakthrough, Valiant demonstrating that the dependency on single-query ANN can be beat using the polynomial methodm resulting in a $O(n^{2-\Omega(\sqrt{\varepsilon})})$ algorithm. This was subsequently improved by [ACW16b] to the current best known $O(n^{2-\tilde{\Omega}(\varepsilon^{1/3})})$ runtime.

Previous reductions between geometric problems to ANN and CP.   Several other works have utilized approximate nearest neighbor (ANN) or closest pair as a subroutine to solve another geometric task. For instance, in [HIM12] they show how one can solve the Minimum Spanning Tree (MST) problem via calls to a dynamic ANN algorithm (which they implement via LSH). For low-dimensional space, [Aga+90] reduces MST to closest pair, though they incur incurring an exponential dependence on the demonstrate d. For high-dimensional space, [ACW16a] reduces MST to CP, achieving runtimes comparable to their CP algorithm. For large constant approximations of EMD, [AS14a] use an ANN subroutine to obtain to obtain a $O(C^{\log(n)})$ approximation in $n^{1+O(1/C)}$ time, although this approach is now dominated by the spanner with fast min-cost flow solver approach described above.

4 Preliminaries

Notation.   For any integer $n \geqslant 1$, we write $[n] = \{1, 2, \ldots, n\}$, and for two integers $a, b \in \mathbb{Z}$, write $[a..b] = \{a, a+1, \ldots, b\}$. For $a, b \in \mathbb{R}$ and $\varepsilon \in (0, 1)$, we use the notation $a = (1 \pm \varepsilon)b$ to denote the containment of $a \in [(1 - \varepsilon)b, (1 + \varepsilon)b]$. We will use boldface symbols to represent random variables and functions, and non-boldface symbols for fixed values (potentially realizations of these random variables) for instance $\boldsymbol{f}$ vs. f. We use $\tilde{O}(\cdot), \tilde{\Omega}(\cdot)$ notation to hide polylog factors in n, d, ε.

Choice of ℓ_p norm, and magnitude of the coordinates. For the remainder of the paper, we assume our point sets X, Y lie in $(\mathbb{R}^d, \ell_1)$. This is without loss of generality, since for any $p \in [1, 2]$ there exists an embedding of ℓ_p^d to $\ell_1^{d'}$ with approximately preserves all distances to a factor of $(1 + \varepsilon)$, where $d' = O(d \log(1/\varepsilon)/\varepsilon^2)$ [JS82]. Naturally, our results equally apply to any underlying metric which can be similarly embedded into ℓ_1.

Without loss of generality, we assume our pointsets X, Y are contained in $[1, \Phi]^d$ for some integer $\Phi \geqslant 0$. For simplicity and to avoid additional notation, we also allow Φ to be a bound on the aspect ratio of A, B, namely, $1 \leqslant \|x - y\|_1 \leqslant \Phi$ for any $(x, y) \in X \times Y$ (note that, a priori, the aspect ratio would be at most $\Phi \cdot d$. Note that we can easily remove and match duplicate points in X, Y in $O(nd)$ time in pre-processing, and may therefore assume that $X \cap Y = \varnothing$. In Section A, we will show a reduction to $\Phi = \mathrm{poly}(n, d)$ bounded aspect ratio. We assume further withlot loss of generality that $\Phi = \omega(1)$.

For two sets $X, Y \subset \mathbb{R}^d$ with the same number of points and $p \in [1, 2]$, we write $\mathsf{EMD}_{\ell_p}(X, Y)$ to denote the EMD between X, Y with respect to the $(\mathbb{R}^d, \ell_p)$. Since ℓ_p can be embedded into ℓ_1 with small distortion, we often focus on the case of ℓ_1 and write $\mathsf{EMD}(X, Y) = \mathsf{EMD}_{\ell_1}(X, Y)$

Approximate Closest Pair. We now formally define the approximate closest pair problem, which is the key problem that we will reduce to. Note that we allow our closest pair oracle to be randomized and succeed with probability at least $2/3$. Of course, this may be boosted to an arbitrary $1 - 1/\mathrm{poly}(n)$ success probability via independent repetition.

Definition 4.1 (Approximate Closest Pair). *Fix any $\varepsilon \in (0, 1)$. Then the $(1+\varepsilon)$-approximate closest pair (CP) problem as the following. Given $A, B \subseteq (\mathbb{R}^d, \ell_1)$ with $|A|, |B| \leqslant n$ and a parameter $\varepsilon > 0$, return $(a, b) \in A \times B$ such that $||a - b||_1 \leqslant (1 + \varepsilon) \cdot \min_{x \in A, y \in B} ||x - y||_1$. We allow a CP algorithm to be randomized, so long as it succeeds with probability at least $2/3$.*

5 Bi-Lipschitz Quadtree Embedding

In this section, we give an approximate tree embedding embedding of our dataset with stronger guarantees than what is possible via standard probabilistic tree embeddings. Our construction is an post-hoc modification to the probabilistic tree embeddings from [AIK08b].

Specifically, [AIK08b] shows how to construct a probabilistic tree embedding for $(\mathbb{R}^d, \ell_1)$, such that each pairwise distance is lower bounded with high probability, and such that each pairwise distance is upper bounded in *expectation*. While these guarantees are asymmetric in nature, they are sufficient for approximating EMD up to a poly-logarithmic factor. However, for our purpose we will need a stronger bi-Lipschitz guarantee—namely, all pairwise distances should be preserved up to a poly-logarithmic factor with a tree metric. Unfortunately, for a general point set $X \subset (\mathbb{R}^d, \ell_1)$, a bi-Lipschitz tree embeddings with poly-logarithmic distortion does not exist;[8] however, we show how to slightly perturb points in X so that such tree embeddings exist while simultaneously ensuring that the Earth Mover's distance on the perturbed points remains the same up to a $(1+\varepsilon)$-factor. Importantly, this perturbation is performed *post-hoc*, after the tree embedding itself has been generated, as it depends on the tree embedding itself.

We consider the following modified notation for expressing EMD for arbitrary supply and demand vectors over differing metrics.

---

[8]For example, for large n, the n-cycle can be approximated by a large circle in $\mathbb{R}^2$, and n-cycle metrics are known to require $\Omega(n)$ distortion when mapped into trees [RR98].

Definition 5.1. *Let $X = ([n], \mathsf{d})$ be a metric, and let $V \subset \mathbb{R}^n$ be the subspace of vectors $b \in \mathbb{R}^n$ which satisfy $\langle b, \mathbb{1}_n \rangle = 0$. For any $b \in V$, the value $\mathsf{EMD}_X(b) \in \mathbb{R}_{\geqslant 0}$ is defined as follows:*

- *Let $b_+, b_- \in \mathbb{R}^n_{\geqslant 0}$ divide positive and negative parts of b, so $b_+ - b_- = b$.*

- *We consider the complete bipartite graph $[n] \times [n]$, where the cost of the edge $(i,j) \in [n] \times [n]$ is $\mathsf{d}(i, x_j)$.*

- *A feasible flow $\gamma \in \mathbb{R}^{n \times n}_{\geqslant 0}$ is one which satisfies $\sum_{j=1}^n \gamma_{ij} = (b_+)_i$ for all $i \in [n]$ and $\sum_{i=1}^n \gamma_{ij} = (b_-)_j$ for all $j \in [n]$.*

We let

$$\mathsf{EMD}_X(b) = \min_{\substack{\gamma \in \mathbb{R}^{n \times n}_{\geqslant 0} \\ \textit{feasible flow}}} \sum_{i=1}^n \sum_{j=1}^n \gamma_{ij} \cdot \mathsf{d}(i,j),$$

and will often refer to the flow γ realizing $\mathsf{EMD}_X(b)$ as the minimizing feasible flow.

One useful property of $\mathsf{EMD}_X(b)$ as taken above is that it forms a norm over the set of vectors b whose coordinates sum to zero (i.e. demand is equal to supply).

Fact 5.2. *Let $X = ([n], \mathsf{d})$ be a metric, and let $\Lambda = \{ b \in \mathbb{R}^d \mid \langle b, \mathbb{1}_d \rangle = 0 \}$. Then $\mathsf{EMD}_X(b)$ is a norm over Λ.*

We will often write $\|b\|_{\mathsf{EMD}} := \mathsf{EMD}_X(b)$ when the metric X is fixed. We often consider a size-n subsets $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$. In these cases, we implicitly associate $\{x_1, \ldots, x_n\}$ with the set $[n]$, and refer to X as the metric with $\mathsf{d}(i,j) = \|x_i - x_j\|_1$. We use the following probabilistic tree embedding from [AIK08b], which embeds subsets $X \subset (\mathbb{R}^d, \ell_1)$ of bounded aspect ratio into a probabilistic (rooted) tree via a collection of nested and randomly shifted grids. This allows to translate between the Earth mover's distance among points in X, EMD_X, to the Earth mover's distance over a tree metric. Recall that a tree metric is specified by a tree T with weighted edges, and defines the metric on the vertices by letting the distance $\mathsf{d}_T(i,j)$ be the length of the path between i and j in the tree.

Lemma 5.3 ([AIK08b]). *Fix $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$ whose non-zero pairwise distances lie in $[1, \Phi]$. There is a distribution $\mathcal{T} = \mathcal{T}(X)$ which satisfies the following guarantees:*

- *Every tree in $\mathcal{T}$ is rooted and has n leaves which are associated with $x_1, \ldots, x_n$.*

- *The depth of every tree T in $\mathcal{T}$ is $h = O(\log \Phi)$, with the root at depth 0 and leaves at h.*

- *Every edge connecting a vertex at depth ℓ to a vertex at depth $\ell + 1$ has the same weight $\Phi/2^\ell$.*

Furthermore, distances in a randomly sampled tree $\mathbf{T} \sim \mathcal{T}$ satisfy:

- *For any $i, j \in [n]$ and any $\delta \in (0, 1)$, $\mathsf{d}_{\mathbf{T}}(i,j) \geqslant \|x_i - x_j\|_1 / O(\log(1/\delta))$ with probability at least $1 - \delta$ over $\mathbf{T} \sim \mathcal{T}$.*

- *For any $b \in V$, let $\gamma \in \mathbb{R}^{n \times n}_{\geqslant 0}$ denote the min-cost flow realizing $\mathsf{EMD}_X(b)$. Then,*

$$\mathop{\mathbf{E}}_{\mathbf{T}} \left[\sum_{i=1}^n \sum_{j=1}^n \gamma_{ij} \cdot \mathsf{d}_{\mathbf{T}}(i,j) \right] \leqslant O(\log \Phi) \cdot \mathsf{EMD}_X(b).$$

15

The main algorithm of the section is in Figure 2, which allows us to use the probabilistic tree to perturb the points. The perturbation ensures that the tree is a bi-Lipschitz embedding of the perturbed point set, and that the Earth mover's distance in the original point set and the Earth mover's distance in the perturbed point set does not change significantly.

Algorithm EMBED-AND-PERTURB(X, ε).

- **Input**: a set of points $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$ whose pairwise distances lie in $\{0\} \cup [1, \Phi]$, as well as an accuracy parameter $\varepsilon \in (0, 1)$.

- **Output**: A tuple $(\mathbf{Y}, \mathbf{T})$, where $\mathbf{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\} \subset (\mathbb{R}^{d+d'}, \ell_1)$ with $d' = O(\log n)$, and $\mathbf{T}$ is a tree metric with n leaves associated with the points in $\mathbf{Y}$.

The algorithm proceeds as follows:

1. Sample a tree $\mathbf{T} \sim \mathcal{T}$ from the distribution specified in Lemma 5.3.

2. For every node $v \in \mathbf{T}$ at depth ℓ, sample $\boldsymbol{z}_v \sim [0, \varepsilon \cdot \Phi/2^\ell]^{d'}$.

3. For each $i \in [n]$, let $\boldsymbol{v}_0, \ldots, \boldsymbol{v}_h$ denote the root-to-leaf path in $\mathbf{T}$, where $\boldsymbol{v}_h$ is leaf i. Let $\boldsymbol{y}_i \in \mathbb{R}^{d+d'}$ where the first d coordinates are x_i, and the final d' coordinates are set to $(1/d') \sum_{i=0}^{h} \boldsymbol{z}_{v_i}$.

Figure 2: The algorithm EMBED-AND-PERTURB which receives a set of vectors X, and produces a perturbed set of vectors and a bi-Lipschitz tree embedding for the perturbed vectors.

Lemma 5.4. *There is a constant $c > 0$ such that for any $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$ with distances in $[1, \Phi]$, the algorithm* EMBED-AND-PERTURB$(X, c\varepsilon/\log \Phi)$ *outputs a tuple $(\mathbf{T}, \mathbf{Y})$ which, for any $b \in V$, satisfies the following with probability at least 0.9:*

- *The set $\mathbf{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\} \subset (\mathbb{R}^{d+O(\log n)}, \ell_1)$ satisfies $\mathsf{EMD}_X(b) \leqslant \mathsf{EMD}_\mathbf{Y}(b) \leqslant (1 + \varepsilon) \cdot \mathsf{EMD}_X(b)$.*

- *The set $\mathbf{Y}$ is embedded in the tree metric $\mathbf{T}$ of n leaves at depth $O(\log \Phi)$ which achieves bi-Lipschitz distortion $O(\log n \log \Phi/\varepsilon)$.*

Furthermore, the algorithm runs in time $O(nd \log n \log \Phi)$.

The guarantees of the algorithm in Figure 2 are proven in Claims 5.6 and 5.7. First, that the tree $\mathbf{T}$ is a bi-Lipschitz embedding of the output vectors $\mathbf{Y}$, and then, that for a fixed vector $b \in V$, the Earth mover's distance $\mathsf{EMD}_X(b)$ is approximately the same as $\mathsf{EMD}_\mathbf{Y}(b)$. In order to do the analysis, we consider the following three events, where each occurs with high probability and imply the desired guarantees:

- $\boldsymbol{\mathcal{E}}_1$: This event depends on the draw of $\mathbf{T} \sim \mathcal{T}$ from Lemma 5.3. It occurs whenever every $i, j \in [n]$ satisfies $\|x_i - x_j\|_1/O(\log n) \leqslant \mathsf{d}_\mathbf{T}(i, j)$. Setting $\delta = 1/n^{10}$ in Lemma 5.3 and a union bound, $\boldsymbol{\mathcal{E}}_1$ occurs with probability $1 - o(1)$.

- $\mathcal{E}_2$: We consider a fixed vector $b \in V$, and we let $\gamma \in \mathbb{R}_{\geqslant 0}^{n \times n}$ be the flow which realizes $\mathsf{EMD}_X(b)$. This event depends on the draw of $\mathbf{T} \sim \mathcal{T}$ from Lemma 5.3, and occurs whenever

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \cdot \mathsf{d_T}(i, j) \leqslant O\left(\log \Phi\right) \cdot \mathsf{EMD}_X(b).$$

  From Lemma 5.3 and Markov's inequality, this event occurs with probability at least 0.99.

- $\mathcal{E}_3$: Fix a draw of $\mathbf{T}$, and this event depends on the draws $\boldsymbol{z_v}$ from Line 2 of Figure 2. It occurs whenever the following holds for every $i, j \in [n]$: we let $\boldsymbol{v}_0, \ldots, \boldsymbol{v}_h$ and $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_h$ be the root-to-leaf paths to leaves i and j in $\mathbf{T}$, respectively; then,

$$\left\| \frac{1}{d'} \sum_{\ell=0}^{h} (\boldsymbol{z_{v_\ell}} - \boldsymbol{z_{u_\ell}}) \right\|_1 = \Theta(\varepsilon) \cdot \mathsf{d_T}(i, j).$$

  We prove that event $\mathcal{E}_3$ occurs with high probability below.

Claim 5.5. *Fix any tree* $\mathbf{T}$ *from Lemma 5.3 and any* $i, j \in [n]$. *Let* $\boldsymbol{v}_0, \ldots, \boldsymbol{v}_h$ *and* $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_h$ *be the root-to-leaf paths to* i *and* j *in* $\mathbf{T}$, *respectively. Then,*

$$\left\| \frac{1}{d'} \sum_{\ell=0}^{h} (\boldsymbol{z_{v_\ell}} - \boldsymbol{z_{u_\ell}}) \right\|_1 = \Theta(\varepsilon) \cdot \mathsf{d_T}(i, j)$$

with probability at least $1 - 1/n^{10}$ *over the draw of* $\boldsymbol{z_{v_\ell}}, \boldsymbol{z_{u_\ell}} \sim [0, \varepsilon \cdot \Phi/2^\ell]^{d'}$ *for* $\ell \in \{0, \ldots, h\}$.

Proof. For any $k \in [d']$, we let $\mathbf{S}_k \in \mathbb{R}$ denote the k-th coordinate of the vector $(1/d') \sum_{\ell=0}^{h} (\boldsymbol{z_{v_\ell}} - \boldsymbol{z_{u_\ell}})$. We may equivalently write the random variable

$$\mathbf{S}_k = \frac{1}{d'} \sum_{\ell=0}^{h} (\boldsymbol{z_{v_\ell}} - \boldsymbol{z_{u_\ell}})_k = \frac{1}{d'} \sum_{\ell=0}^{h} \mathbf{1}\{\boldsymbol{v}_\ell \neq \boldsymbol{u}_\ell\} \cdot \frac{\varepsilon \cdot \Phi}{2^\ell} (\boldsymbol{a}_k - \boldsymbol{b}_k),$$

where $\boldsymbol{a}_k, \boldsymbol{b}_k \sim [0, 1]$. Since the random variable $\boldsymbol{a}_k - \boldsymbol{b}_k$ is symmetric about the origin, the Khintchine inequality implies that

$$\mathbf{E}\left[|\mathbf{S}_k|\right] = \Theta(1/d') \cdot \left(\sum_{\ell=0}^{h} \mathbf{1}\{\boldsymbol{v}_\ell \neq \boldsymbol{u}_\ell\} \cdot \frac{\varepsilon^2 \cdot \Phi^2}{2^{2\ell}} \right)^{1/2} = \Theta(\varepsilon/d') \cdot \mathsf{d_T}(i, j),$$

where the last equality comes from the draw of $\mathbf{T}$ in Lemma 5.3, since $\mathsf{d_T}(i, j)$ is exactly $2 \sum_{\ell=0}^{h} \mathbf{1}\{\boldsymbol{v}_\ell \neq \boldsymbol{u}_\ell\} \cdot \Phi/2^\ell$. Similarly, $|\mathbf{S}_k|$ is bounded by $\Theta(\varepsilon/d') \cdot \mathsf{d_T}(i, j)$ as well, the event follows from standard concentration inequalities on sums of $d' = \Omega(\log n)$ independent and bounded random variables. $\square$

Claim 5.6. *Execute* EMBED-AND-PERTURB(X, ε) *and assume events* $\mathcal{E}_1$ *and* $\mathcal{E}_3$ *hold. Then, every* $i, j \in [n]$ *satisfies*

$$\varepsilon \cdot \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1 \leqslant \mathsf{d_T}(i, j) \leqslant O\left(\log n\right) \cdot \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1.$$

Proof. We lower bound

$$\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1 \geqslant \left\| \frac{1}{d'} \sum_{\ell=0}^{h} (\boldsymbol{z}_{\boldsymbol{v}_\ell} - \boldsymbol{z}_{\boldsymbol{u}_\ell}) \right\|_1 = \Omega(\varepsilon) \cdot \mathsf{d}_{\mathbf{T}}(i,j),$$

from $\boldsymbol{\mathcal{E}}_3$, and furthermore, we can upper bound using both $\boldsymbol{\mathcal{E}}_1$ and $\boldsymbol{\mathcal{E}}_3$

$$\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1 = \|x_i - x_j\|_1 + \left\| \frac{1}{d'} \sum_{\ell=0}^{h} (\boldsymbol{z}_{\boldsymbol{v}_\ell} - \boldsymbol{z}_{\boldsymbol{u}_\ell}) \right\|_1 \leqslant O(\log(n)) \cdot \mathsf{d}_{\mathbf{T}}(i,j) + O(\varepsilon) \cdot \mathsf{d}_{\mathbf{T}}(i,j),$$

which is at most $O(\log n) \cdot \mathsf{d}_{\mathbf{T}}(i,j)$, since $\varepsilon \in (0,1)$. $\qquad\square$

Claim 5.7. *For any vector $b \in V$, execute* EMBED-AND-PERTURB(X, ε) *and assume $\boldsymbol{\mathcal{E}}_2$ and $\boldsymbol{\mathcal{E}}_3$ hold. Then,*
$$\mathsf{EMD}_X(b) \leqslant \mathsf{EMD}_{\mathbf{Y}}(b) \leqslant (1 + O(\varepsilon \log \Phi)) \cdot \mathsf{EMD}_X(b).$$

Proof. The upper bound, $\mathsf{EMD}_X(b) \leqslant \mathsf{EMD}_{\mathbf{Y}}(b)$, occurs with probability 1 because every $i, j \in [n]$ always satisfies $\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1 \geqslant \|x_i - x_j\|_1$. This means that any feasible flow will have its cost be larger in $\mathbf{Y}$ than in X. In order to prove the upper bound, let $\gamma \in \mathbb{R}_{\geqslant 0}^{n \times n}$ denote the feasible flow which realizes $\mathsf{EMD}_X(b)$, so

$$\mathsf{EMD}_X(b) = \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \cdot \|x_i - x_j\|_1.$$

We can upper bound the cost of this flow by first applying the conditions of $\boldsymbol{\mathcal{E}}_3$, followed by the conditions of $\boldsymbol{\mathcal{E}}_2$:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \cdot \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1 \leqslant \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \left(\|x_i - x_j\|_1 + \Theta(\varepsilon) \cdot \mathsf{d}_{\mathbf{T}}(i,j) \right)$$

$$= \mathsf{EMD}_X(b) + \Theta(\varepsilon) \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \cdot \mathsf{d}_{\mathbf{T}}(i,j)$$

$$\leqslant (1 + O(\varepsilon \log \Phi)) \cdot \mathsf{EMD}_X(b).$$

$\qquad\square$

Reduction to Polynomial Aspect Ratio. We show that it suffices to consider a bounded aspect ratio $\Phi = \mathrm{poly}(nd)$.

Lemma 5.8. *There exists a randomized algorithm which runs in $O(nd + n \log n)$ time and has the following guarantees:*

- **Input**: *A set $X = \{x_1, \dots, x_n\} \subset (\mathbb{R}^d, \ell_1)$ and an integer vector $b \in V$.*

- **Output**: *A collection of points $\mathbf{Y}_1, \dots, \mathbf{Y}_t \subset \mathbb{R}^{d + O(\log n)}$ and a corresponding set of supply/demand integer vectors $b_1, \dots, b_t \in V$.*

18

With probability at least 0.9*, the aspect ratio of each* $\mathbf{Y}_i$ *is at most* $\mathrm{poly}(nd)$*, and*

$$\left| \mathsf{EMD}_X(b) - \sum_{i=1}^{t} \mathsf{EMD}_{\mathbf{Y}_i}(b_i) \right| \leqslant \varepsilon \cdot \mathsf{EMD}_X(b).$$

The proof of this lemma is in Appendix A, and proceeds along the following lines. First, we show that we may obtain a polynomial approximation to $\mathsf{EMD}_X(b)$ in near-linear time. The polynomial approximation is enough for us to randomly partition the instance into parts of small diameter, such that it suffices to independently solve each part. Finally, we add a small amount of noise to each point in each part, which ensures that the minimum pairwise distance is lower bounded, without significantly affecting the value of the Earth Mover's Distance.

6 Computing EMD via Multiplicative Weights Update

In this section, we present a template of our algorithm for computing $\mathsf{EMD}_{\ell_1}(X, Y)$, for input multi-sets $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$ from $\mathbb{R}^d$. We seek an algorithm which obtains a $(1 + \varepsilon)$-approximation for $\varepsilon \in (0, 1)$. Following Section 5, we may assume the following facts about our input:

- The points $x_1, \ldots, x_n, y_1, \ldots, y_n \in \mathbb{R}^d$ all have non-zero pairwise distances in $[1, \Phi]$, where we may assume Φ is a power of 2 and at most $\mathrm{poly}(nd)$.

- There is a rooted tree T of depth $h = O(\log \Phi)$ with root at depth 0 and $2n$ leaves (associated with each point of X and Y) at depth h, and every edge from depth ℓ to $\ell + 1$ has weight $\Phi/2^\ell$ (which will be an integer).

- The tree T gives a bounded-distortion embedding for pairs of points in X and Y; i.e., for parameters $\mathsf{D}_l = \varepsilon / \log \Phi$ and $\mathsf{D}_u = O(\log n)$, every pair $i, j \in [n]$ satisfies

$$\mathsf{D}_l \cdot \|x_i - y_j\|_1 \leqslant \mathsf{d}_T(i, j) \leqslant \mathsf{D}_u \cdot \|x_i - y_j\|_1. \tag{11}$$

Note moreover, by Lemma 5.8, we may assume a polynomial aspect ratio: namely $\Phi = \mathrm{poly}(n, d)$. Recall that a bottom-up greedy assignment from X to Y in T gives a matching which realizes $\mathsf{EMD}_T(X, Y)$. This bottom-up greedy matching may be computed in $O(n \log \Phi)$ time, and because all pairwise distances satisfy (11), we find a number $t_0 \in \mathbb{R}_{\geqslant 0}$ which satisfies, for $\mathsf{D}_T = O(\log n \log \Phi / \varepsilon)$,

$$t_0 \leqslant \mathsf{EMD}_{\ell_1}(X, Y) \leqslant \mathsf{D}_T \cdot t_0.$$

In this section, we show that there exist a randomized algorithm satisfies the following: if $\mathsf{EMD}_{\ell_1}(X, Y) \geqslant (1 + 3\varepsilon)t$, it can produce a "certificate" that $\mathsf{EMD}_{\ell_1}(X, Y)$ is larger than t (which is correct except with a negligible probability); if it cannot find such certificate, it outputs "fail." This guarantee is enough to obtain a $(1 + O(\varepsilon))$-approximation to $\mathsf{EMD}_{\ell_1}(X, Y)$ up to an additional $O(\log \mathsf{D}_T / \varepsilon)$ factor in the running time, by starting at $t = t_0$ and outputting the smallest $t \leqslant \mathsf{D}_T t_0$ where the algorithm outputs "fail." First, we establish the notion of the certificate to prove that $\mathsf{EMD}_{\ell_1}(X, Y) \geqslant t$.

Definition 6.1 (Consistent Rounding of Distances)**.** *Fix* $\varepsilon \in (0, 1)$ *and the set of points* $X = \{x_1, \ldots, x_n\}$ *and* $Y = \{y_1, \ldots, y_n\}$ *with distances in* $[1, \Phi]$*. Consider the two matrices:*

- **Rounded Distances.** *The matrix $\psi \in \{0, \dots, \log \Phi/\varepsilon\}^{n \times n}$ is such that $i, j \in [n]$ satisfies $(1 + \varepsilon)^{\psi_{ij}} \leqslant \|x_i - y_j\|_1 \leqslant (1 + \varepsilon)^{\psi_{ij}+1}$.*

- **Up/Down Rounding**. *The boolean matrix $S \in \{0, 1\}^{n \times n}$, to be specified, which will decide how to round the distance of pair (i, j).*

We define $C = C(S)$ to be the $n \times n$ cost matrix of rounded distances, where $C_{ij} = (1 + \varepsilon)^{\psi_{ij} - 2 \cdot S_{ij}}$, and note that

$$\left| \frac{C_{ij}}{\|x_i - y_j\|_1} - 1 \right| \leqslant 2\varepsilon.$$

Definition 6.2 (Cost Certificate). *For an $n \times n$ matrix C (from Definition 6.1), we let $\alpha, \beta \in \mathbb{R}^n$ be two vectors, and for $t \geqslant t_0$, we let*

$$\Gamma_t = \left\{ (\alpha, \beta) \in \mathbb{R}^n \times \mathbb{R}^n : \frac{1}{t} \cdot \left(\sum_{i=1}^n \alpha_i - \sum_{j=1}^n \beta_j \right) - \max_{i,j} \frac{|\alpha_i - \beta_j|}{C_{ij}} > 0 \right\}.$$

Claim 6.3. *If $\Gamma_t \neq \varnothing$, then $\mathsf{EMD}_{\ell_1}(X, Y) \geqslant (1 - 2\varepsilon)t$.*

Proof. Suppose $(\alpha, \beta) \in \Gamma_t$ and define

$$(\alpha', \beta') := \left(\max_{i,j \in [n]} \frac{|\alpha_i - \beta_j|}{||a_i - b_j||} \right)^{-1} \cdot (\alpha, \beta).$$

Then, $(\alpha', \beta') \in \Gamma_t$, it is a feasible dual solution, and moreover $\max_{i,j \in [n]} \frac{\alpha'_i - \beta'_j}{||x_i - y_j||_1} = 1$. Thus $\max_{i,j \in [n]} \frac{\alpha'_i - \beta'_j}{C_{i,j}} \geqslant (1 - 2\varepsilon)$. Thus, $\mathsf{EMD} \geqslant \sum_{i \in [n]} \alpha_i - \beta_i > (1 - 2\varepsilon)t$ as desired. $\qquad \square$

Algorithmic Plan. From Claim 6.3, an algorithm which can find a point (α, β) guaranteed to lie in Γ_t has a certificate that $\mathsf{EMD}_{\ell_1}(X, Y) \geqslant (1 - 2\varepsilon)t$. Our plan is to: (i) either we find that $\mathsf{EMD}_{\ell_1}(X, Y) \leqslant (1 + 3\varepsilon)t$, or (ii) execute a procedure which outputs a pair (α, β) such that, except with a small error probability, $(\alpha, \beta) \in \Gamma_t$. A sub-quadratic time algorithm may store a pair $(\alpha, \beta) \in \mathbb{R}^n \times \mathbb{R}^n$ (as it requires $2n$ numbers to specify). Note, however, that the polytope Γ_t is defined by $2n^2$ inequality constraints, which means it is not at all clear how to verify whether $(\alpha, \beta) \in \Gamma_t$ (much less find one). We will find such a pair (α, β) with a multiplicative weights update rule. The algorithm will be iterative, and each iteration will require us to solve the task in Figure 3.

Oracle Task $\text{CERTIFY}(\lambda, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$.

- **Input**: A distribution λ supported on $[n] \times [n] \times \{-1, 1\}$, and a matrix $C \in \mathbb{R}^{n \times n}$ with $C_{ij} = (1 \pm \varepsilon) \cdot \|x_i - y_j\|$.

- **Output**: Either the algorithm outputs a pair $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{Z}^n \times \mathbb{Z}^n$ satisfying the following: We let $\boldsymbol{v} = (1/t)(\sum_{i=1}^n \boldsymbol{\alpha}_i - \sum_{j=1}^n \boldsymbol{\beta}_j)$, and we should have

$$\sum_{i=1}^n \sum_{j=1}^n \sum_{\sigma \in \{-1,1\}} \lambda_{i,j,\sigma} \frac{\sigma(\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j)}{C_{ij}} \leqslant \boldsymbol{v} - \gamma_{\mathsf{gap}}, \qquad \max_{i,j} \frac{|\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j|}{C_{ij}} \leqslant \mathsf{K}, \qquad \text{and} \qquad |\boldsymbol{v}| \leqslant \mathsf{K},$$

  or the algorithm outputs "Fail".

If $\text{CERTIFY}(\lambda, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ returns "Fail", then $\text{EMD}_{\ell_1}(X, Y) \leqslant (1 + 3\varepsilon)t$.

Figure 3: The certification oracle task for each iteration.

Lemma 6.4 below demonstrates an algorithm that can solve the CERTIFY task by taking only $\tilde{O}(n/\delta)$ samples. For technical reasons, we will not be able to sample exactly from λ, but instead will have access to samples from a distribution λ' which is very close to λ in total variational distance. Namely, we can guarantee $\|\lambda' - \lambda\|_{\text{TV}} \leqslant 1/\text{polylog}(n)$ (for a sufficiently large polylogarithmic factor). Nevertheless, we show that we can still solve CERTIFY even given access only to the approximate distribution λ'.

Note that Lemma 6.4 has a small failure probability δ, and requires $\tilde{O}(n/\delta)$ samples to succeed. However, since the procedure will only be called $\text{polylog}(n)$ times, it will suffice to set $\delta = 1/\text{polylog}(n)$, and thus we will require only $\tilde{O}(n)$ samples for every call to CERTIFY.

Lemma 6.4. *For any $\varepsilon, \delta \in (0, 1)$, suppose that*

$$\gamma_{\mathsf{gap}} \leqslant \frac{\varepsilon \cdot \mathsf{D}_l}{2h} \qquad \text{and} \qquad \mathsf{K} \geqslant \mathsf{D}_u \cdot \mathsf{D}_T,$$

where $\mathsf{D}_T = O(\varepsilon^{-1} \cdot \log n \cdot \log \Phi)$. Moreover, suppose that we have sample access to a distribution λ' such that $\|\lambda' - \lambda\|_{\text{TV}} = o\left(\frac{\varepsilon \delta \mathsf{D}_\ell}{h \mathsf{D}_u}\right)$. Then there is a randomized algorithm which, with probability $1 - \delta$, solves $\text{CERTIFY}(\lambda, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ using $\tilde{O}(n/\delta)$ samples from λ' and running in time $O(nd \log \Phi)$.

Proof. We begin by assuming we have sample access directly to λ, and later correct for the fact that we only have sample access to λ'. Consider the supply and demand vectors $\mu \in \mathbb{R}^n$ and $\nu \in \mathbb{R}^n$ which are given by:

$$\mu_i = t \sum_{j=1}^n \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \qquad \text{and} \qquad \nu_j = t \sum_{i=1}^n \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}},$$

where we use $\lambda_{i,j,\sigma} \in [0, 1]$ to be the probability that the tuple (i, j, σ) is sampled under λ. Notice that the flow $\gamma_{ij} = t(\lambda_{i,j,1} - \lambda_{i,j,-1})/C_{ij}$ is by definition a feasible flow for $\text{EMD}_{\ell_1}(\mu, \nu)$ with cost

$$\sum_{i=1}^n \sum_{j=1}^n |\gamma_{ij}| \cdot \|x_i - y_j\|_1 = t \sum_{i=1}^n \sum_{j=1}^n |\lambda_{i,j,1} - \lambda_{i,j,-1}|(1 + \varepsilon) \leqslant (1 + \varepsilon)t.$$

21

Here we have two cases: (i) $\mathsf{EMD}_{\ell_1}(\mathbf{1} - \mu, \mathbf{1} - \nu) < \varepsilon t$; (ii) $\mathsf{EMD}_{\ell_1}(\mathbf{1} - \mu, \mathbf{1} - \nu) \leqslant \varepsilon t$. If we are in case (i), by the triangle inequality for the Earth Mover's Distance,

$$\mathsf{EMD}(\mathbf{1}, \mathbf{1}) \leqslant \mathsf{EMD}(\mu, \nu) + \mathsf{EMD}(\mathbf{1} - \mu, \mathbf{1} - \nu) < (1 + \varepsilon)t + \varepsilon t = (1 + 2\varepsilon)t.$$

Since distance in T can be lower bounded by distances in ℓ_1 (as per (11)), in case (ii) we have:

$$\mathsf{EMD}_T(\mathbf{1} - \mu, \mathbf{1} - \nu) \geqslant \mathsf{D}_l \cdot \mathsf{EMD}_{\ell_1}(\mathbf{1} - \mu, \mathbf{1} - \nu) \geqslant \mathsf{D}_l \cdot \varepsilon t,$$

and hence there exists a depth of the tree $\ell \in [h]$ which satisfies:

$$\sum_{v \in T_\ell} \frac{\Phi}{2^\ell} \left| |X_v| - |Y_v| - t \sum_{\substack{i \in [n] \\ x_i \in v}} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} + t \sum_{\substack{j \in [n] \\ y_j \in v}} \sum_{i=1}^{n} \sum_{\sigma \in \{-1,1\}} \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \right| \geqslant \frac{\mathsf{D}_l \cdot \varepsilon t}{h}. \qquad (12)$$

Algorithmically, we iterate over tree depths $\ell \in [h]$ and check whether Equation (12) holds. If *Equation* (12) never holds, then we return "Fail". As we proved, if *Equation* (12) never holds, then it must be that we are in case (i), thus $\mathsf{EMD}(\mathbf{1}, \mathbf{1}) \leqslant (1 + 2\varepsilon)t$, ensuring correctness. Naively, checking whether Equation (12) holds required knowledge of the values $\lambda_{i,j,\sigma}$. Later, we will show that this step can be implemented only using samples from λ.

Suppose, instead, that we find $\ell \in [h]$ satisfying Equation (12). We now consider the following algorithm to find a setting of (α, β).

1. Let $(\boldsymbol{i}_1, \boldsymbol{j}_1, \boldsymbol{\sigma}_1), \ldots, (\boldsymbol{i}_s, \boldsymbol{j}_s, \boldsymbol{\sigma}_s) \sim \lambda$ denote s independent samples from the distribution.

2. For each $v \in T_\ell$ and $k \in [s]$, define $\mathbf{Z}_v^k = (\mathbf{1}\{x_{\boldsymbol{i}_k} \in v\} - \mathbf{1}\{y_{\boldsymbol{j}_k} \in v\}) \dfrac{\boldsymbol{\sigma}_k}{C_{\boldsymbol{i}_k \boldsymbol{j}_k}}$

3. For each $v \in T_\ell$, let $\boldsymbol{w}_v \in \{-1, 1\}$ be given by:

$$\boldsymbol{w}_v = \mathrm{sign}\left(|X_v| - |Y_v| - \frac{t}{s} \sum_{k=1}^{s} \mathbf{Z}_v^k \right).$$

4. Finally, for every $v \in T_\ell$ and for every $i, j \in [n]$ with $x_i, y_j \in v$, we set

$$\boldsymbol{\alpha}_i = \boldsymbol{\beta}_j = \frac{\Phi}{2^\ell} \cdot \boldsymbol{w}_v.$$

For each $v \in T_\ell$, let $s_v \in \{-1, 1\}$ denote the sign which realizes the absolute value in the right-hand side of (12) for v. First, suppose if we always had $\boldsymbol{w}_v = s_v$. In this case, we would obtain the "optimal" values $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$, which satisfy that for every $i \in [n]$ with $x_i \in v$ and $j \in [n]$ with $y_j \in v$, we have

$$\boldsymbol{\alpha}_i^* = \boldsymbol{\beta}_j^* = s_v \cdot \frac{\Phi}{2^\ell},$$

thus, in this case, we may simplify (12) as

$$\sum_{v \in T_\ell} \frac{\Phi}{2^\ell} \cdot s_v \left(|X_v| - |Y_v| - t \sum_{i \in [n]} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} (\mathbf{1}\{x_i \in v\} - \mathbf{1}\{y_j \in v\}) \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \right)$$

22

$$= \sum_{i=1}^{n} \boldsymbol{\alpha}_i^* - \sum_{j=1}^{n} \boldsymbol{\beta}_j^* - t \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \lambda_{i,j,\sigma} \cdot \sigma \cdot \frac{\boldsymbol{\alpha}_i^* - \boldsymbol{\beta}_j^*}{C_{ij}} \geqslant \frac{\mathsf{D}_l \cdot \varepsilon t}{h},$$

and in particular,

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma \in \{-1,1\}} \lambda_{i,j,\sigma} \cdot \sigma \cdot \frac{\boldsymbol{\alpha}_i^* - \boldsymbol{\beta}_j^*}{C_{ij}} \leqslant \frac{1}{t} \left(\sum_{i=1}^{n} \boldsymbol{\alpha}_i^* - \sum_{j=1}^{n} \boldsymbol{\beta}_j^* \right) - \frac{\varepsilon \cdot \mathsf{D}_l}{h}.$$

$$= \boldsymbol{v} - \gamma_{\mathsf{gap}}$$

So the idealized values $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ satisfy the first constraint of the Lemma. Thus, in order for the actual values $(\boldsymbol{\alpha}, \boldsymbol{\beta})$'s we compute to satisfy the first constraint, it will suffices to set the values of $\boldsymbol{w}_v$ such that

$$\left| \sum_{v \in T_\ell} \frac{\Phi}{2^\ell} \cdot (\boldsymbol{w}_v - s_v) \left(|X_v| - |Y_v| - t \sum_{\substack{i,j \in [n] \\ \sigma \in \{-1,1\}}} (\mathbf{1}\{x_i \in v\} - \mathbf{1}\{y_j \in v\}) \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} \right) \right| \tag{13}$$

$$= \eta \leqslant \frac{\mathsf{D}_l \cdot \varepsilon t}{2h}$$

We proceed to bound the error η. First, note the following:

Claim 6.5. *For any $v \in T_\ell$, and $i, j \in [n]$, if $x_i \in v$ and $y_j \notin v$ (or vice-versa), then $C_{ij} \geqslant \frac{(1-\varepsilon)\Phi}{2^\ell \mathsf{D}_u}$.*

Proof. By the guarantees of the tree embedding [11], and using that $C_{ij} = (1 \pm \varepsilon)\|x_i - y_j\|_1$, we have that

$$C_{ij} \geqslant (1 - \varepsilon)\|x_i - y_j\|_1 \geqslant \frac{(1-\varepsilon)}{\mathsf{D}_u} \mathsf{d}_T(i,j) \geqslant \frac{(1-\varepsilon)\Phi}{2^\ell \mathsf{D}_u}$$

Where the last inequality holds because x_i, y_j split at or above level ℓ in the tree. $\square$

Further, note that any i, j with $x_i \in v$ and $y_j \in v$ (or $x_i \notin v$ and $y_j \notin v$) contributes zero to the sum $\sum_{\substack{i,j \in [n] \\ \sigma \in \{-1,1\}}} (\mathbf{1}\{x_i \in v\} - \mathbf{1}\{y_j \in v\}) \lambda_{i,j,\sigma} \cdot \sigma / C_{ij}$, and thus we can restrict the sum of pairs i, j that are split by a vertex v. To proceed, we define

$$\eta_v = \frac{t\Phi}{2^\ell} \left| \sum_{\substack{i,j \in [n] \\ \sigma \in \{-1,1\}}} (\mathbf{1}\{x_i \in v\} - \mathbf{1}\{y_j \in v\}) \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}} - \frac{1}{s} \sum_{k=1}^{s} \mathbf{Z}_v^k \right| \tag{14}$$

and note that we can bound η via $\eta \leqslant 2 \sum_{v \in T_\ell} \eta_v$, where here we are using the observation that for any vectors $a, b, w \in \mathbb{R}^n$ where $w_i = \mathrm{sign}(b_i)$, we have $|\langle w, a \rangle - \|a\|_1| \leqslant 2\|a - b\|_1$. Since $\mathbf{E}\left[\mathbf{Z}_v^k\right] = \sum_{\substack{i,j \in [n] \\ \sigma \in \{-1,1\}}} (\mathbf{1}\{x_i \in v\} - \mathbf{1}\{y_j \in v\}) \frac{\lambda_{i,j,\sigma} \cdot \sigma}{C_{ij}}$, by Jensen's inequality:

$$\mathbf{E}\left[\eta_v\right] \leqslant \frac{t\Phi}{2^\ell} \cdot \sqrt{\mathbf{Var}\left(\frac{1}{s} \sum_{i=1}^{s} \mathbf{Z}_v^k \right)}$$

23

Using Claim 6.5, we proceed to bound the above via:

$$\mathbf{Var}\left(\mathbf{Z}_v^k\right) \leqslant \sum_{\substack{i,j\in[n]\\\sigma\in\{-1,1\}\\|\{x_i,x_j\}\cap v|=1}} 4 \cdot \frac{\lambda_{i,j,\sigma}}{C_{ij}^2} \leqslant \left(\frac{2^{\ell+2}\mathsf{D}_u}{\Phi}\right)^2 \cdot \sum_{\substack{i,j\in[n]\\\sigma\in\{-1,1\}\\|\{x_i,x_j\}\cap v|=1}} \lambda_{i,j,\sigma}$$

Thus

$$\mathbf{E}\left[\eta_v\right] \leqslant \frac{4\mathsf{D}_u t}{\sqrt{s}} \cdot \sqrt{\sum_{\substack{i,j\in[n]\\\sigma\in\{-1,1\}\\|\{x_i,x_j\}\cap v|=1}} \lambda_{i,j,\sigma}}$$

We have

$$\mathbf{E}\left[\eta\right] \leqslant \sum_{v\in T_\ell} \mathbf{E}\left[\eta_v\right] \leqslant \frac{4\mathsf{D}_u t}{\sqrt{s}} \cdot \sum_{v\in T_\ell} \sqrt{\sum_{\substack{i,j\in[n]\\\sigma\in\{-1,1\}\\|\{x_i,x_j\}\cap v|=1}} \lambda_{i,j,\sigma}}$$

$$\leqslant \sqrt{2n}\frac{4\mathsf{D}_u t}{\sqrt{s}} \cdot \left(\sum_{v\in T_\ell}\left|\sum_{\substack{i,j\in[n]\\\sigma\in\{-1,1\}\\|\{x_i,x_j\}\cap v|=1}} \lambda_{i,j,\sigma}\right|\right)^{1/2} \leqslant \sqrt{4n}\frac{4\mathsf{D}_u t}{\sqrt{s}}$$

Where we first used the fact that $\sum_{i=1}^m \sqrt{|a_i|} \leqslant \sqrt{m}(\sum_{i=1}^n |m_i|)^{1/2}$ for any sequence $a_1,\ldots,a_m$, which follows from Cauchy Schwartz, and we next used that each $\lambda_{i,j,\sigma}$ appears in the sum for at most two vertices $v \in T_\ell$, and lastly that λ is a distribution. Therefore, setting $s = O\left(\left(\frac{h\mathsf{D}_u}{\varepsilon\mathsf{D}_l}\right)^2 \frac{n}{\delta^2}\right) = \tilde{O}(n/\delta^2)$, we obtain $\mathbf{E}\left[\eta\right] \leqslant \frac{\varepsilon\delta\mathsf{D}_l t}{h100}$, and the desired bound on η holds with probability at least $1 - \delta$ by Markov's inequality.

Checking Equation (12). The previous analysis shows that the error η_v in Equation (14) satisfies $\eta_v \leqslant \frac{\mathsf{D}_l \cdot \varepsilon t}{2h}$. Thus, given $\ell \in [h]$, we can check if Equation (12) holds for $\varepsilon' = 3/2\varepsilon$, which suffices to obtain

$$\mathsf{EMD}(\mathbf{1},\mathbf{1}) \leqslant \mathsf{EMD}(\mu,\nu) + \mathsf{EMD}(\mathbf{1}-\mu,\mathbf{1}-\nu) < (1+\varepsilon)t + \varepsilon't \leqslant (1+3\varepsilon)t,$$

ensuring correctness in the "Fail" case.

Sampling from λ' instead of λ. Next, we claim that the same sampling procedure works when our samples $(\boldsymbol{i}_1,\boldsymbol{j}_1,\boldsymbol{\sigma}_1),\ldots,(\boldsymbol{i}_s,\boldsymbol{j}_s,\boldsymbol{\sigma}_s)$ are drawn from λ' instead of λ. This modifies the distribution of the variables $\mathbf{Z}_v^k$ accordingly. Call the modified variables $\widehat{\mathbf{Z}}_v^k$. To analyze the modified variables,

note that we can bound the error of η coming from the new variables $\widehat{\mathbf{Z}}_v^k$ via

$$
\begin{aligned}
\mathbf{E}\left[\eta\right] \leqslant \sum_v \mathbf{E}\left[\eta_v\right] \leqslant & \frac{t\Phi}{2^\ell} \left| \sum_{\substack{i,j\in[n] \\ \sigma\in\{-1,1\}}} (\mathbf{1}\left\{x_i \in v\right\} - \mathbf{1}\left\{y_j \in v\right\}) \frac{\lambda'_{i,j,\sigma} \cdot \sigma}{C_{ij}} - \frac{1}{s}\sum_{k=1}^s \widehat{\mathbf{Z}}_v^k \right| \\
& + \frac{t\Phi}{2^\ell} \left| \sum_{\substack{i,j\in[n] \\ \sigma\in\{-1,1\}}} (\mathbf{1}\left\{x_i \in v\right\} - \mathbf{1}\left\{y_j \in v\right\}) \frac{|\lambda'_{i,j,\sigma} - \lambda_{i,j,\sigma}| \cdot \sigma}{C_{ij}} \right| \\
\leqslant & \frac{t\Phi}{2^\ell} \sum_v \left| \sum_{\substack{i,j\in[n] \\ \sigma\in\{-1,1\}}} (\mathbf{1}\left\{x_i \in v\right\} - \mathbf{1}\left\{y_j \in v\right\}) \frac{\lambda'_{i,j,\sigma} \cdot \sigma}{C_{ij}} - \frac{1}{s}\sum_{k=1}^s \widehat{\mathbf{Z}}_v^k \right| + 8 \cdot t \cdot \mathsf{D}_u \|\lambda - \lambda'\|_{\mathrm{TV}}
\end{aligned}
\tag{15}
$$

Where in the last line, via Claim 13 we used that $C_{i,j} \geqslant \frac{(1-\varepsilon)\Phi}{2^\ell \mathsf{D}_u}$ for every (i,j,v) where $\mathbf{1}\left\{x_i \in v\right\} - \mathbf{1}\left\{y_j \in v\right\} \neq 0$. We also used that each term $|\lambda'_{i,j,\sigma} - \lambda_{i,j,\sigma}|$ appears exactly twice in the sum over all vertices $v \in T_\ell$, allowing us to bound the contribution of those terms by $2 \cdot \|\lambda - \lambda'\|_{\mathrm{TV}}$. Now note that the first term on the last line of (15) can be bounded by $\frac{t\Phi}{2^\ell} \sum_v \cdot \sqrt{\mathbf{Var}\left(\frac{1}{s}\sum_{i=1}^s \widehat{\mathbf{Z}}_v^k\right)}$, and thus by $\sqrt{4n}\frac{4\mathsf{D}_u t}{\sqrt{s}}$ via the same sequence of inequalities as above. Using that $\|\lambda' - \lambda\|_{\mathrm{TV}} = o\left(\frac{\varepsilon\mathsf{D}_\ell}{h\mathsf{D}_u}\right)$, we obtain $\mathbf{E}\left[\eta\right] \leqslant \frac{\varepsilon\mathsf{D}_l t}{h50\delta}$ as before, which is only a factor of 2 larger than the earlier computation using λ, and the statement again follows by Markov's inequality.

Finally, for the last two constraints, note that $|\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j| \leqslant d_T(i,j) \leqslant \mathsf{D}_u \|x_i - y_j\|_1 \leqslant \mathsf{D}_u(1+\varepsilon)C_{ij}$, and therefore, we have $\sum_{i=1}^n \boldsymbol{\alpha}_i - \sum_{j=1}^n \boldsymbol{\beta}_j \leqslant \mathsf{D}_u \cdot \mathsf{EMD}_{\ell_1}(X,Y) \leqslant \mathsf{D}_u \mathsf{D}_T t$, where the last inequality holds because we know $t_0 \leqslant t \leqslant \mathsf{D}_T \cdot t_0$ and moreover $\mathsf{EMD}_{\ell_1}(X,Y) \leqslant \mathsf{D}_T \cdot t_0$. Thus, $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ satisfy the desired constraints.

$\square$

6.1 Multiplicative Weights Update from a Bounded Class of Distributions

As we explain now, the class of distributions λ which we will consider will be parametrized over dual variables (α, β), which are the values which get updated as the algorithm proceeds and evolve the distribution λ. In order to efficiently sample from λ, it will be important for us to appropriately round to an accuracy χ and maintain the signs of the dual variables (α, β).

Definition 6.6 (Rounded Dual Variables). *Fix $\chi \in (0,1)$ and a pair $(\alpha, \beta) \in \mathbb{Z}^n \times \mathbb{Z}^n$.*

- **Rounded Duals.** *D is the $n \times n$ matrix with entries in $\{0\} \cup \{(1+\chi)^h : h \in \mathbb{Z}_{\geqslant 0}\}$ satisfying*

$$D_{ij} \leqslant |\alpha_i - \beta_j| \leqslant (1+\chi) \cdot D_{ij}.$$

- **Sign Pattern.** *P is the $n \times n$ matrix with $P_{ij} = \mathrm{sign}(\alpha_i - \beta_j)$.*

We now define a bounded class of distributions λ which depend on C, D and P. In the remainder of this section, we first define the class of distributions λ. Then, we show that a multiplicative weights update rule will produce a sequence of distributions λ in this class. Furthermore, iteratively solving the certification task on these distributions is enough to find a pair $(\alpha, \beta) \in \Gamma_t$.

Definition 6.7 (Class of Distributions). *Fix $\eta \in (0, 1)$ and $S \in \{0, 1\}^{n \times n}$. Define matrices D, P as in Definition 6.6 and matrix $C = C(S)$ as in Definition 6.1. Define weights $w_{i,j,\sigma} \geqslant 0$ for $i, j \in [n]$ and $\sigma \in \{-1, 1\}$:*

$$w_{i,j,\sigma} = \exp(\eta \cdot (\sigma \cdot P_{ij}) \cdot D_{ij}/C_{ij}).$$

The distribution $\lambda(\eta, w) = \lambda(\eta, C, D, P)$ is that which samples with probability proportionally to w, and $\mathcal{D}(\eta)$ is the class of distributions specified by η, C, D and P.

In the remainder of the section, we show the multiplicative weights update (MWU) algorithm equipped with an oracle for $\textsc{Certify}(\lambda, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ on distributions $\lambda \in \mathcal{D}$ can be used to find a certificate $(\alpha, \beta) \in \Gamma_t$. In particular, we analyze the iterative procedure in Figure 4.

Thanks to Lemma 6.4, if we can design an efficient algorithm to sample from distributions in $\mathcal{D}$, then we can implement the oracle for $\textsc{Certify}(\lambda, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ efficiently, which in turn makes the MWU algorithm efficient.

Multiplicative Weights Update. There is a fixed matrix of rounded distances C satisfying $C_{ij} = (1 \pm \varepsilon) \cdot \|x_i - y_j\|$.

1. Initialize $(\alpha^1, \beta^1) \in \mathbb{Z}^n \times \mathbb{Z}^n$ to all zero. This specifies $v^1 = 0$, matrix D^1 to all zero, and P^1 to all one, and weights $w^1_{i,j,1} = w^1_{i,j,-1} = 1$.

2. Repeat for rounds $r = 1, \ldots, \mathsf{R} - 1$:

   - Let $\lambda^r = \lambda(\eta, w^r) = \lambda(\eta, C, D^r, P^r)$ as in Definition 6.7.
   - If $\textsc{Certify}(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ outputs "Fail", return "Fail".
   - Else, let $(\tilde{\alpha}^r, \tilde{\beta}^r)$ be the output of $\textsc{Certify}(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$.
   - Update $(\alpha^{r+1}, \beta^{r+1}) = (\alpha^r, \beta^r) + (\tilde{\alpha}^r, \tilde{\beta}^r)$. This specifies $v^{r+1} = v^r + \tilde{v}^r$, matrices D^{r+1} and P^{r+1}, and weights w^{r+1}.

3. Output $1/\mathsf{R} \cdot (\alpha^{\mathsf{R}}, \beta^{\mathsf{R}})$

If this algorithm returns "Fail", then $\mathrm{EMD}(X, Y) \leqslant (1 + 3\varepsilon)t$. Else, if the algorithm returns (α, β), then $(\alpha, \beta) \in \Gamma_t$.

Figure 4: The Multiplicative Weights Update Rule.

Lemma 6.8. *Run the multiplicative weights algorithm in Figure 4 with parameters:*

$$\eta \leqslant \frac{\gamma_{\mathsf{gap}}}{100 \cdot \mathsf{K}^2} \qquad \mathsf{R} \geqslant \frac{\ln(2n^2)}{100 \cdot \eta \gamma_{\mathsf{gap}}} \qquad and \qquad \chi \leqslant \frac{\gamma_{\mathsf{gap}}}{100 \cdot \mathsf{RK}}.$$

Suppose it returns (α, β) at line 3. Then, $(\alpha, \beta) \in \Gamma_t$.

Observation 2. Assume correctness of CERTIFY. Thanks to Lemma 6.4, if the algorithm in Figure 4 returns "Fail", then $\mathrm{EMD}(X, Y) \leqslant (1 + 3\varepsilon)t$. Thanks to Lemma 6.8, if if the algorithm Figure 4 returns (α, β), then $(\alpha, \beta) \in \Gamma_t$.

6.2 Proof of Lemma 6.8

We first make the following observations, which dictate how the weights w^r evolve in the multiplicative weights update procedure. The goal will be to upper bound the sum of all weights in the algorithm, so we let $W^r = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma \in \{-1, 1\}} w_{i,j,\sigma}^r$.

Observation 3. For any $i, j \in [n]$ and $\sigma \in \{-1, 1\}$. Then,

$$w_{i,j,\sigma}^{r+1} \leqslant w_{i,j,\sigma}^r \cdot \exp\left(\eta\chi \cdot 2(r+1)\mathsf{K}\right) \cdot \exp\left(\eta\sigma(\tilde{\alpha}_i^r - \tilde{\beta}_j^r)/C_{ij}\right).$$

Proof of Observation 3. Here, we expand the definition of $w_{i,j,\sigma}^{r+1}$, and we use the fact $P_{ij}^{r+1} D_{ij}^{r+1} \leqslant \alpha_i^{r+1} - \beta_j^{r+1} + \chi D_{ij}^{r+1}$.

$$w_{i,j,\sigma}^{r+1} = \exp\left(\eta(\sigma \cdot P_{ij}^{r+1}) D_{ij}^{r+1}/C_{ij}\right) \leqslant \exp\left(\eta\sigma(\alpha_i^{r+1} - \beta_j^{r+1})/C_{ij}\right) \cdot \exp\left(\eta\chi D_{ij}^{r+1}/C_{ij}\right).$$

We may now expand the first exponential in the right-hand side, and have:

$$\exp\left(\eta\sigma(\alpha_i^{r+1} - \beta_j^{r+1})/C_{ij}\right) = \exp\left(\eta\sigma(\alpha_i^r - \beta_j^r)/C_{ij}\right) \cdot \exp\left(\eta\sigma(\tilde{\alpha}_i^r - \tilde{\beta}_j^r)/C_{ij}\right)$$

$$\leqslant w_{i,j,\sigma}^r \cdot \exp\left(\eta\chi D_{ij}^r/C_{ij}\right) \cdot \exp\left(\eta\sigma(\tilde{\alpha}_i^r - \tilde{\beta}_j^r)/C_{ij}\right),$$

where the last inequality also uses the fact $P_{ij}^r D_{ij}^r$ is at least $\alpha_i^r - \beta_j^r + \chi D_{ij}^r$. The final inequality follows from combining both of the above, and noting the fact that both D_{ij}^r/C_{ij} and D_{ij}^{r+1}/C_{ij} are at most $(r+1)\mathsf{K}$ (since they are at worst $r+1$ sums of terms which are smaller than K). $\qquad\square$

We may now use the choice of $(\tilde{\alpha}^r, \tilde{\beta}^r)$ being the solution to CERTIFY$(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ in the analysis of how the weights evolve. The following lemma follows the traditional analysis of the multiplicative weights update method up to the small error incurred from rounding the dual variables.

Observation 4. If $(\tilde{\alpha}^r, \tilde{\beta}^r)$ satisfies the guarantees of CERTIFY$(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ (in Figure 3) and $\eta \leqslant 1/(100\mathsf{K})$. Then,

$$W^{r+1} \leqslant \exp\left(\eta\chi \cdot r\mathsf{K} + \tilde{v}^r - \gamma_{\mathsf{gap}} + O(\eta^2\mathsf{K}^2)\right) \cdot W^r$$

Proof. Here, we will use the fact that when the guarantees of Figure 3. In particular, we may take second-degree Taylor expansion of the exponential function, and have

$$\mathop{\mathbf{E}}_{(i,j,\sigma) \sim \lambda^r}\left[\exp\left(\eta\boldsymbol{\sigma}(\tilde{\alpha}_i^r - \tilde{\beta}_j^r)/C_{ij} - \tilde{v}^r + \gamma_{\mathsf{gap}}\right)\right]$$

$$\leqslant 1 + \eta\left(\mathop{\mathbf{E}}_{(i,j,\sigma) \sim \lambda^r}\left[\frac{\boldsymbol{\sigma}(\tilde{\alpha}_i^r - \tilde{\beta}_j^r)}{C_{ij}}\right] - \tilde{v}^r + \gamma_{\mathsf{gap}}\right) + O(\eta^2\mathsf{K}^2)$$

$$\leqslant 1 + O(\eta^2\mathsf{K}^2) \leqslant \exp\left(O(\eta^2\mathsf{K}^2)\right).$$

27

We apply Observation 3

$$W^{r+1} = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma} w_{i,j,\sigma}^{r+1} \leqslant \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\sigma} w_{i,j,\sigma}^{r} \cdot \exp\left(\eta\chi \cdot 2(r+1)\mathsf{K}\right) \cdot \exp\left(\eta\sigma(\tilde{\alpha}_i^r - \tilde{\beta}_j^r)/C_{ij}\right)$$

$$\leqslant W^r \exp\left(\eta\chi \cdot 2(r+1)\mathsf{K} + \tilde{v}^r - \gamma_{\mathsf{gap}}\right) \underset{(\boldsymbol{i,j,\sigma}) \sim \lambda^r}{\mathbf{E}} \left[\exp\left(\eta\boldsymbol{\sigma}(\tilde{\alpha}_{\boldsymbol{i}}^r - \tilde{\beta}_{\boldsymbol{j}}^r)/C_{\boldsymbol{ij}} - \tilde{v}^r + \gamma_{\mathsf{gap}}\right)\right]$$

$$\leqslant W^r \cdot \exp\left(\eta\chi \cdot 2(r+1)\mathsf{K} + \tilde{v}^r - \gamma_{\mathsf{gap}} + O(\eta^2\mathsf{K}^2)\right).$$

$\square$

From here, we may conclude the proof of Lemma 6.8. In particular, after all R rounds, we have:

$$W^{\mathsf{R}} \leqslant 2n^2 \cdot \left(\eta\chi \cdot 2\mathsf{R}^2\mathsf{K} + \eta \sum_{r=1}^{\mathsf{R}-1} \tilde{v}^r - (\mathsf{R}-1)\gamma_{\mathsf{gap}} + O(\mathsf{R}\eta^2\mathsf{K}^2)\right),$$

and this implies every $i, j \in [n]$ satisfies

$$\frac{|\alpha_i - \beta_j|}{C_{ij}} = \frac{1}{\mathsf{R}} \cdot \frac{|\alpha_i^{\mathsf{R}} - \beta_j^{\mathsf{R}}|}{C_{ij}} \leqslant \frac{1}{\eta \cdot \mathsf{R}} \ln\left(w_{i,j,1}^{\mathsf{R}} + w_{i,j,-1}^{\mathsf{R}}\right) \leqslant \frac{\ln(2n^2)}{\eta \cdot \mathsf{R}} + \chi\mathsf{RK} + v - \gamma_{\mathsf{gap}}/2 + O(\eta\mathsf{K}^2).$$

The parameter settings of Lemma 6.8 implies

$$\max_{i,j} \frac{|\alpha_i - \beta_j|}{C_{ij}} \leqslant v - \gamma_{\mathsf{gap}}/10 \leqslant v,$$

which means $(\alpha, \beta) \in \Gamma_t$.

6.3 Proof of the Main Theorem

We now state our main theorem, which provides a $(1 + \varepsilon)$ approximation. Note that, by standard independent repetition and outputting the median of estimates, we can boost the failure probability to $1 - 1/\text{poly}(n)$ with a $O(\log n)$ increase in the runtime.

Theorem 5 (Main Theorem, formal). *Suppose that there exists an algorithm for $(1+\varepsilon)$-approximate closest pair (Definition 4.1) on $(\mathbb{R}^d, \ell_1)$ with running time $T_{CP}(n, \varepsilon) = n^{2-\phi(\varepsilon)} + O(nd)$ and success probability at least $2/3$. Then, there exists an algorithm that computes a $(1 + O(\varepsilon))$-approximation to $\mathsf{EMD}(X, Y)$ over $(\mathbb{R}^d, \ell_p)$ for any $p \in [1, 2]$ in time $\tilde{O}(n^{2-\Omega(\phi(\varepsilon))} + nd)$ and success probability at least $2/3$.*

Proof. Let $X, Y \subseteq ([1, \Phi]^d, \ell_1)$ our size-n input sets. Observation 2 guarantees that: if the MWU algorithm in Figure 4 returns $(\alpha, \beta) \in \Gamma_t$, thus certifying $\mathsf{EMD}(X, Y) \geqslant (1 - 2\varepsilon)t$ by Claim 6.3; else, if MWU returns "fail", then $\mathsf{EMD}(X, Y) \leqslant (1 + 3\varepsilon)t$.

Moreover, Theorem 6 ensures that the MWU algorithm in Figure 4 can be implemented in time $\tilde{O}(n^{2-\Omega(\phi)} + nd)$ and with success probability $1 - 1/\text{poly}(n, \log \Phi)$. Thus, we can run an exponential search for $\mathsf{EMD}(X, Y)$ with parameter $t_k = (1+\varepsilon)^k$ and return the smallest t_k such that the MWU algorithm in Figure 4 returns "Fail". Since Lemma 5.8 guarantees $\Phi \leqslant \text{poly}(nd)$, then $\text{polylog}(nd)$ steps of binary search suffice. $\square$

7 Sampling from the Distribution λ via Closest Pair

The main result of this section is the following.

Theorem 6 (Subquadratic implementation of MWU). *Fix $\varepsilon, \phi > 0$. Suppose that there exists a randomized algorithm for $(1+\varepsilon)$-approximate closest pair on $(\mathbb{R}^d, \ell_1)$ with running time $T_{CP}(n, \varepsilon) = n^{2-\phi}$ and success probability at least $2/3$. Then, is possible to implement the multiplicative weights update algorithm in Figure 4 in time $\tilde{O}(n^{2-\Omega(\phi)} + nd)$ with success probability $1 - 1/\mathrm{poly}(n, \log \Phi)$.*

Proof. Notice that we can boost the success probability of the CP algorithm to $1 - 1/\mathrm{poly}(n)$ by re-running it $O(\log n)$ times and returning the closest pair among those runs. Since we are going invoke the CP algorithm at most $\mathrm{poly}(n)$ times, we can condition on the event that the CP algorithm never fails.

Recall that $\mathsf{D}_l = \varepsilon / \log \Phi$, $\mathsf{D}_u = O(\log n)$ and $h = O(\log \Phi)$, as defined at the beginning of Section 6. According to Lemma 6.4, we can choose

$$\gamma_{\mathsf{gap}} = \frac{\varepsilon \cdot \mathsf{D}_l}{2h} \qquad \text{and} \qquad \mathsf{K} = \mathsf{D}_u \cdot \mathsf{D}_T, \qquad \text{where} \qquad \mathsf{D}_T = O(\varepsilon^{-1} \cdot \log n \cdot \log \Phi).$$

Likewise, according Lemma 6.8, we can choose

$$\eta = \frac{\gamma_{\mathsf{gap}}}{100 \cdot \mathsf{K}^2} \qquad \mathsf{R} = \frac{\ln(2n^2)}{100 \cdot \eta \gamma_{\mathsf{gap}}} \qquad \text{and} \qquad \chi = \frac{\gamma_{\mathsf{gap}}}{100 \cdot \mathsf{RK}}.$$

Thus, $\mathsf{K}, \mathsf{R}, \eta^{-1}, \chi^{-1}, \gamma_{\mathsf{gap}}^{-1} \leqslant \mathrm{poly}(\log(n\Phi), \varepsilon^{-1})$. Finally, define $\tau := n^{1+\phi/2}$.

Algorithm. Upfront, we sample a set $S \subseteq X \times Y$ of size $n^{2-\phi/8}$ uniformly at random. By Lemma 6.8, to implement the MWU algorithm in Figure 4, we just need to correctly implement $\mathrm{CERTIFY}(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ for $r = 1 \ldots \mathsf{R}$. By Lemma 6.4, we can implement $\mathrm{CERTIFY}(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ as long as we can sample from a distribution λ' such that $\|\lambda' - \lambda^r\|_{\mathrm{TV}} = o\left(\frac{\varepsilon \delta \mathsf{D}_\ell}{h \mathsf{D}_u}\right) = o\left(\frac{\varepsilon^2}{\log^2 n \cdot \log^2 \Phi}\right)$. In what follows, we set $\delta < 1/(100R) = 1/\mathrm{polylog}(n)$ so that we can guarantee correctness of the $\mathrm{CERTIFY}$ procedure over all R rounds (recall that δ is the failure probability of the $\mathrm{CERTIFY}$ procedure). Recall that $\mathrm{CERTIFY}(\lambda^r, \gamma_{\mathsf{gap}}, \mathsf{K}, \delta)$ returns $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ satisfying

$$|\boldsymbol{\alpha}_i - \boldsymbol{\beta}_j| \leqslant C_{ij} \cdot \mathsf{K} \leqslant \mathrm{poly}(n, \Phi),$$

hence throughout the execution of MWU we have $\|\boldsymbol{\alpha}\|_\infty, \|\boldsymbol{\alpha}\|_\infty \leqslant \mathrm{poly}(n, \Phi)$. Recall that $\lambda^r = \lambda^r(\eta, C, D^r, P^r)$ belongs to the class of distributions $\mathcal{D}(\eta)$ defined in Definition 6.7. In the following, we show how to sample from λ^r by leveraging Lemmas 7.2, 7.8 and 7.9.

Combining Lemma 7.2 and Lemma 7.8 we obtain the following algorithm.

ConstantSampler

- **Input**: Sets $X', Y' \subseteq (\mathbb{R}^d, \ell_1)$ of size m, a set $S \subseteq X' \times Y'$ of size $m^{2-\rho\phi}$ for some constant $\rho \in (0, 1/2)$, a matrix $P \in \{\pm 1\}^{m \times m}$, and a matrix $D \in \mathbb{R}^{m \times m}$ with all equal entries.

- **Output**: n samples from $\lambda = \lambda(\eta, C(S), D, P)$, where $C(S)$ is defined as in Definition 6.1.

ConstantSampler is correct with high probability, as long as S τ-shatters the collection $\{L_t(X', Y')\}$. ConstantSampler runs in time $\tilde{O}(m^{2-\Omega(\phi)})$.

Notice that the matrices C, D, P fed as input to `ConstantSampler` are represented implicitly according to Definitions 6.1 and 6.6, allowing for subquadratic running time.

Then, we invoke Lemma 7.9, which yields the following algorithm.

`ArbitrarySampler`

- **Input**: Sets $X, Y \subseteq (\mathbb{R}^d, \ell_1)$ of size n, a set $S \subseteq X \times Y$ of size $n^{2-\phi/8}$, dual solutions $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{Z}^{2n}$, a matrix $P \in \{\pm 1\}^{n \times n}$, and a matrix $D \in \mathbb{R}^{n \times n}$ defined as in Definition 6.6.

- **Output**: $n \cdot \log^z(n)$ samples from λ' such that $|\lambda' - \lambda|_{TV} = o\left(\frac{\varepsilon^2 \delta^2}{\log^2 n \cdot \log^2 \Phi}\right)$, where $\lambda = \lambda(\eta, C(S), D, P)$, $C(S)$ is defined as in Definition 6.1, and $z > 0$ is any constant.

`ArbitrarySampler` calls

$$\texttt{ConstantSampler}(X_i, Y_i, S \cap (X_i \times Y_i), P|_{X_i \times Y_i}, D|_{X_i \times Y_i})$$

for $i \in [k]$, where $\{X_i \times Y_i\}_{i \in [k]}$ is a collection of disjoint combinatorial rectangles in $X \times Y$ of size $\sqrt[4]{n} \times \sqrt[4]{n}$ such that $D|_{X_i \times Y_j}$ has all equal entries. `ArbitrarySampler` succeeds with high probability, as long as `ConstantSampler` is correct on all these calls. Finally, `ArbitrarySampler` runs in time $\tilde{O}(n^{2-\Omega(\phi)})$.

Denote with $\{X_i^r \times Y_i^r\}_{i \in [k]}$ the combinatorial rectangles defined by `ArbitrarySampler` at round r. We would like to prove that S τ-shatters the collection

$$\left\{ L_t(X_i^r, Y_i^r) \, : \, r \in [\mathsf{R}], i \in [k] \text{ and } t = 1 \ldots \varepsilon^{-1} \cdot \log \Phi \right\}. \tag{16}$$

If that is the case, then: (i) the set $S \cap (X_i^r \times Y_i^r)$ fed to `ConstantSampler` has the correct size; (ii) `ConstantSampler` is correct, with high probability. To this end, we invoke Corollary 8.3.

Tracking dependencies. In order to verify the hypotheses of Corollary 8.3, we need to track how the objects defined in our algorithm depend on the random choice of S. When talking about dependencies, we consider all random variables besides S fixed.

Using the notation of Corollary 8.3, we have $D = S$, $A = [n] \times [n]$ and $p^r = \lambda^r$. The state σ_r corresponds to the variables $(\boldsymbol{\alpha}^r, \boldsymbol{\beta}^r)$, which, in turn, define the matrices P^r and D^r. The partition function

$$f_r : [n] \times [n] \to \left\{ L_t(X_i^r, Y_i^r) \, : \, i \in [k] \text{ and } t = 1 \ldots \varepsilon^{-1} \cdot \log \Phi \right\} \perp \{\perp\}$$

corresponds to the assignment of pairs into combinatorial rectangles (or, if $f(i, j) = \perp$, the fact that the pair (i, j) is not assigned to any rectangle. Notice that f_r depends on the state σ^r (through D^r). The state $\sigma_{r+1} = (\boldsymbol{\alpha}^{r+1}, \boldsymbol{\beta}^{r+1})$ depends on $\sigma_r = (\boldsymbol{\alpha}^r, \boldsymbol{\beta}^r)$ as well as the the $\tilde{O}(n)$ samples returned by `ArbitrarySampler` at the previous step. Since the value of $\lambda^r(i, j, \sigma)$ only depends on P_{ij}^r and D_{ij}^r, the distribution λ^r only depends on σ^r.

Analysis. Above, we verified that the collection in Equation (16) is shattered by the set S, which implies that `ConstantSampler` is correct whenever `ArbitrarySampler` calls it. Therefore, the correctness of `ArbitrarySampler` is guaranteed by Lemma 7.9. By running CERTIFY with

a sufficiently small parameter $\delta = 1/\text{polylog}(n\Phi)$ we ensure correctness across all R rounds. If CERTIFY is correct, then by Observation 2, the MWU algorithm is also correct.

The running time of the algorithm is $\tilde{O}(n^{2-\Omega(\phi)} + nd)$. In fact, the algorithm runs R $\leqslant$ poly$(\log(n\Phi), \varepsilon^{-1}$ rounds, and in each round it solves CERTIFY using $\tilde{O}(n)$ samples and $O(nd\Phi)$ time (Lemma 6.4). Finally, to compute the $\tilde{O}(n)$ samples, the algorithm ArbitrarySampler uses time $\tilde{O}(n^{2-\phi})$.

$\square$

7.1 From Closest Pair to All Close Pairs Retrieval

Definition 7.1. *Fix any two sets $X, Y \subset \mathbb{R}^d$, such that $\|x - y\|_1 \in [1, \Phi]$ for all $x \in X, y \in Y$. Then for any $t \geqslant 0$, define the level set.*

$$L_t(X,Y) = \{(x,y) \in X \times Y \mid (1+\varepsilon)^{t-1} \leqslant \|a-b\| < (1+\varepsilon)^t\}$$

When the sets X, Y are fixed from context, we will write $L_t = L_t(X, Y)$. Further, we define the prefix set $\mathcal{L}_t = \mathcal{L}_t(X,Y) = \bigcup_{j \leqslant t} L_j(X,Y)$.

Lemma 7.2. *Suppose that there exists an algorithm that solves ε-approximate closest pair in time $T(n, \varepsilon) = n^{2-\phi}$. Let $X, Y \subseteq (\mathbb{R}^n, \ell_1)$ be size-n sets such that $\|x-y\| \in [1, \Phi]$ for each $(x, y) \in X \times Y$. Then, the algorithm FindClosePairs in Figure 7 returns and integer $t \geqslant 0$ along with the prefix set $\mathcal{L}_t$ such that $|\mathcal{L}_t| = \tilde{O}(n^{1+\phi})$ and the level set L_{t+3} satisfies $|L_{t+3}| = \tilde{\Omega}(n^{1+\phi})$. FindClosePairs runs in time $\tilde{O}(n^{2-\Omega(\phi)})$.*

In the remainder of this section, we prove Lemma 7.2.

Algorithm SubsCP(X, Y, η).

1. Let X' (resp. Y') be the set obtained by subsampling X (resp. Y) with rate η.

2. Return a ε-approximate closest pair in $X' \times Y'$, with success probability $1 - n^{-3}$.

Figure 5: Implementation of SubsCP.

Lemma 7.3. *If $|X| = |Y| = n$, SubsCP takes time $\tilde{O}((\eta \cdot n)^{2-\Omega(\phi)})$, with high probability.*

Proof. First, observe that subsampling with rate η can be implemented in time $\tilde{O}(\eta \cdot n) = \tilde{O}((\eta \cdot n)^{2-\Omega(\phi)})$. In fact, it is sufficient to sample a binomial random variable $b \sim \text{Bin}(n, \eta)$ and then sample b distinct elements from X (resp. Y). By standard concentration bounds $b = \tilde{O}(\eta \cdot n)$ with high probability, hence the claimed running time for the subsampling step.

By conditioning on the event $|X'|, |Y'| = \tilde{O}(\eta \cdot n)$, computing a ε-approximate closest pair takes time $\tilde{O}((\eta \cdot n)^{2-\phi})$. Indeed, if $|X'| \neq |Y'|$ we can simply pad the smallest set with dummy (far-from-all-points) vectors. $\square$

Lemma 7.4. *Fix $z \in [\sqrt{n}, n]$. Let $t = $ LastSmallPrefix(X, Y, z), then $|L_{t+3}| \geqslant \frac{z^2}{\log^3 \Phi}$ and $|\mathcal{L}_{t+2}| \leqslant 0.1 \cdot z^2$, with probability $1 - o(1)$.*

Algorithm `LastSmallPrefix`(X, Y, z).

1. Let $S \subseteq X \times Y$ be a set of $\frac{n^2}{z^2} \log \Phi$ i.i.d. uniform samples from $X \times Y$.

2. Let $t + 3 := \min \{s \geqslant 0 \,|\, L_s \cap S \neq \varnothing\}$.

3. Return t.

Figure 6: Implementation of `LastSmallPrefix`.

Proof. For any $s = 1 \ldots \varepsilon^{-1} \cdot \log \Phi$, if $|L_s| < \frac{z^3}{\log^3 \Phi}$ we have

$$\Pr[L_s \cap S \neq \varnothing] \leqslant$$
$$\sum_{(x,y) \in L_s} \Pr[(x, y) \in S] <$$
$$\frac{z^2}{\log^3 \Phi} \cdot \left(1 - \left(1 - \frac{1}{n^2} \right)^{\frac{n^2}{z^2} \log \Phi} \right) =$$
$$\frac{z^2}{\log^2 \Phi} \cdot O\left(\frac{\log \Phi}{z^2} \right) = O\left(\frac{1}{\log^2 \Phi} \right).$$

Thus, by taking a union bound over all $\varepsilon^{-1} \cdot \log \Phi$ possible values of s we have that, with probability $1 - o(1)$, $|L_s| < \frac{z^3}{\log^3 \Phi}$ implies $L_s \cap S = \varnothing$. Thus, $|L_{t+3}| \geqslant \frac{z^2}{\log^3 \Phi}$.

Let s be the smallest such that $\mathcal{L}_s > 0.1 \cdot z^2$. Then,

$$\Pr[\mathcal{L}_s \cap S = \varnothing] \leqslant \left(1 - \frac{0.1 \cdot z^2}{n^2} \right)^{\frac{n^2}{z^2} \log \Phi} = \frac{1}{\Phi^{\Omega(1)}}.$$

Thus, with probability[9] $1 - o(1)$, we must have $t + 3 \leqslant s$, which implies $\mathcal{L}_{t+2} \leqslant 0.1 \cdot z^2$.

$\square$

---

[9]If $\Phi = \omega(1)$ does not hold, we can replace Φ with $\max\{\Phi, n\}$.

Algorithm FindClosePairs(X, Y).

1. Let $z = \sqrt{n^{1+\phi}}$.

2. $t = \texttt{LastSmallPrefix}(X, Y, z)$.

3. Initialize the set $L = \varnothing$.

4. For $i = 1 \ldots \tilde{O}(z^2)$:

    (a) Let $(x, y) = \texttt{SubsCP}(X, Y, 1/z)$.

    (b) If $(x, y) \in \mathcal{L}_t$, add (x, y) to L.

5. **Invariant (I):** L contains all light edges in $\mathcal{L}_t$.

6. Initialize the counter $c : X \cup Y \to \mathbb{Z}$ to all zeros.

7. For $i = 1 \ldots T = \tilde{O}(z)$:

    (a) Let $(x, y) = \texttt{SubsCP}(X, Y, 1/z)$.

    (b) If $(x, y) \in \mathcal{L}_{t+1}$, set $c[a] \leftarrow c[a] + 1$ and $c[b] \leftarrow c[b] + 1$.

8. Define F as the set of $x \in X \cup Y$ with $c(x) \geqslant 0.02 \cdot T$.

9. **Invariant (II):** $\{0.03\text{-frequent vertices}\} \subseteq F \subseteq \{0.01\text{-frequent vertices}\}$.

10. Initialize the set $H = \varnothing$.

11. For $f \in F$:

    (a) If $f \in X$, then add to H all $(f, b) \in \{f\} \times Y$ such that $(f, b) \in \mathcal{L}_t$.

    (b) If $f \in Y$, then add to H all $(a, f) \in X \times \{f\}$ such that $(a, f) \in \mathcal{L}_t$.

12. **Invariant (III):** H contains all heavy edges in $\mathcal{L}_t$.

13. Return $t, L \cup H$.

Figure 7: Implementation of FindClosePairs.

Throughout, fix $z = \sqrt{n^{1+\phi}}$ and t as in FindClosePairs and condition on the high-probability event in Lemma 7.4. Consider the bipartite graph $G = (X \cup Y, \mathcal{L}_{t+1})$. We say that a vertex $x \in X \cup Y$ is *light* if $\deg_{\mathcal{L}_{t+1}}(x) \leqslant 0.5 \cdot z$, and *heavy* otherwise. We say that an edge $(x, y) \in \mathcal{L}_{t+1}$ is light if both its endpoints are light, and heavy otherwise. We say that a vertex $x \in X \cup Y$ is *C-frequent* if

$$\Pr\left[\texttt{SubsCP}(X, Y, 1/z) \text{ returns and edge from } \mathcal{L}_{t+1} \text{ incident to } x\right] \geqslant C \cdot z^{-1}.$$

Lemma 7.5. *Let $(x, y) \in \mathcal{L}_t$ be a light edge. Then, $\Pr[\text{SubsCP}(X, Y, 1/z) \text{ returns } (x, y)] = \Omega(z^{-2})$.*

33

Proof. The probability that both x and y are subsampled (namely $(x, y) \in X' \times Y'$) is exactly z^{-2}. Let $\mathcal{F}$ be this event. Let $\mathcal{E}$ be the event that some other edge $(x, y) \in \mathcal{L}_{t+1}$ belongs to $X' \times Y'$. Since $(x, y) \in \mathcal{L}_t$, if $\mathcal{F}$ happens and $\mathcal{E}$ does not happen, then our closest-pair algorithm correctly returns (x, y). Our goal is now to upper bound the probability of $\mathcal{E}$ conditioned on $\mathcal{F}$.

Let $\mathcal{E}_1$ be the event that there exists an edge (a, b) with $a \neq x$ and $b \neq y$ such that $(a, b) \in X' \times Y'$. Let $\mathcal{E}_2$ be the event that there exists an edge of the form (a, y) (with $a \neq x$) or (x, b) (with $b \neq y$) that belongs to $X' \times Y'$. Clearly, $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$, so, it is sufficient to show that $\Pr[\mathcal{E}_1 \mid \mathcal{F}] + \Pr[\mathcal{E}_2 \mid \mathcal{F}] \leqslant 1 - \Omega(1)$.

The event $\mathcal{E}_1$ is independent of $\mathcal{F}$, so we can bound

$$\Pr\left[\mathcal{E}_1 \mid \mathcal{F}\right] = \Pr[\mathcal{E}_1] \leqslant |\mathcal{L}_{t+1}| \cdot z^{-2} \leqslant 0.5,$$

where the last inequality follows from *Lemma 7.4*.

Since (x, y) is light, then the $\mathcal{L}_{t+1}$-neighborhoods of x and y are small. More precisely, we have $|N_{\mathcal{L}_{t+1}}(b)|, |N_{\mathcal{L}_{t+1}}(a)| \leqslant 0.5 \cdot z$. So,

$$\Pr\left[N_{\mathcal{L}_{t+1}}(a) \cap Y' \neq \varnothing\right] = 1 - \left(1 - \frac{1}{z}\right)^{|N_{\mathcal{L}_{t+1}}(a)|} \leqslant 1 - \left(1 - \frac{1}{z}\right)^{0.5 \cdot z} = 1 - \frac{1}{e^{0.5}} + o(1) < 0.4.$$

and likewise for $\Pr\left[N_{\mathcal{L}_{t+1}}(b) \cap X' \neq \varnothing\right]$. Therefore,

$$\Pr\left[\mathcal{E}_2 \mid \mathcal{F}\right] \leqslant \Pr\left[N_{\mathcal{L}_{t+1}}(a) \cap Y' \neq \varnothing\right] + \Pr\left[N_{\mathcal{L}_{t+1}}(b) \cap X' \neq \varnothing\right] \leqslant 0.2.$$

Finally, $\Pr[\mathcal{E} \mid \mathcal{F}] \leqslant 0.6$, thus

$$\Pr[\mathcal{F} \text{ and } \neg\mathcal{E}] \geqslant \Pr[\mathcal{F}] \cdot \Pr[\neg\mathcal{E} \mid \mathcal{F}] \geqslant 0.4 \cdot z^{-2}.$$

$\square$

Lemma 7.6. *Every heavy vertex is 0.03-frequent.*

Proof. Consider a heavy vertex $x \in X$ (the case $x \in Y$ is symmetric). Let $\mathcal{E}$ be the event "$\texttt{SubsCP}(X, Y, 1/z)$ returns and edge from $\mathcal{L}_{t+1}$ incident to x". We need to prove that $\Pr[\mathcal{E}] \geqslant 0.03 \cdot z^{-1}$. Let $\mathcal{F}$ be the event $x \in X'$. Let $\mathcal{G}$ be the event $\mathcal{L}_{t+2} \cap (X' \backslash \{x\} \times Y') = \varnothing$ Denote with $N_{\mathcal{L}_{t+1}}(x)$ is the neighborhood of x with respect to edges in $\mathcal{L}_{t+1}$. Let $\mathcal{H}$ be the event $N_{\mathcal{L}_{t+1}}(x) \cap Y' \neq \varnothing$. Notice that $\mathcal{F}$ and $\mathcal{G} \cap \mathcal{H}$ are independent. Moreover, $\mathcal{E} \supseteq \mathcal{F} \cap \mathcal{G} \cap \mathcal{H}$. So, $\Pr[\mathcal{E}] \geqslant \Pr[\mathcal{F}] \cdot \Pr[\mathcal{H} \cap \mathcal{G}]$.

By definition of subsampling we have $\Pr[\mathcal{F}] = z^{-1}$. By union bound over all $|\mathcal{L}_{t+2}| \leqslant 0.1 \cdot z^2$ we have

$$\Pr[\neg\mathcal{G}] \leqslant \sum_{(x, y) \in \mathcal{L}_{t+2}} \frac{1}{z} \leqslant 0.1.$$

Since x is heavy we have $|N_{\mathcal{L}_{t+1}}(x)| > 0.5 \cdot z$, so

$$1 - \Pr[\mathcal{H}] = \Pr[\neg\mathcal{H}] = \left(1 - \frac{1}{z}\right)^{|N_{\mathcal{L}_{t+1}}(x)|} < \left(1 - \frac{1}{z}\right)^{0.5 \cdot z} < 0.65.$$

Thus, $\Pr[\mathcal{G} \cap \mathcal{H}] \geqslant \Pr[\mathcal{H}] - \Pr[\neg\mathcal{G}] \geqslant 0.35 - 0.1 = 0.25$ and we have

$$\Pr[\mathcal{E}] \geqslant z^{-1} \cdot 0.25 \geqslant 0.03 \cdot z^{-1}.$$

$\square$

Lemma 7.7. *There are at most $O(z)$ 0.1-frequent vertices.*

Proof. Let $x \in X$ (the case $x \in Y$ is symmetric). First, we observe that if $\deg_{\mathcal{L}_{t+1}}(x) < 0.01 \cdot z$, then x is not 0.01-frequent. Indeed,

$$\Pr\left[\texttt{SubsCP}(X, Y, 1/z) \text{ returns and edge from } \mathcal{L}_{t+1} \text{ incident to } x\right] \leq$$

$$\Pr[x \in X'] \cdot \Pr\left[Y' \cap N_{\mathcal{L}_{t+1}}(x) \neq \varnothing\right] = \Pr[x \in X'] \cdot \left(1 - \left(1 - \frac{1}{z}\right)^{|N_{\mathcal{L}_{t+1}}(x)|}\right) \leq$$

$$\Pr[x \in X'] \cdot \left(1 - \left(1 - \frac{1}{z}\right)^{0.01 \cdot z}\right) < 0.01 \cdot z^{-1}.$$

By Lemma 7.4, we have $|\mathcal{L}_{t+1}| \leq |\mathcal{L}_{t+2}| = O(z^2)$. Thus, a counting argument shows that there are at most $O(z)$ many $x \in X \cup Y$ with $\deg_{\mathcal{L}_{t+1}}(x) \geq 0.01 \cdot z$. $\qquad\square$

In the end, we prove Lemma 7.2.

Proof of Lemma 7.2. Thanks to Lemma 7.4, we have that $|\mathcal{L}_t| = \tilde{O}(z^2) = \tilde{O}(n^{1+\phi})$ and $|L_{t+3}| = \tilde{\Omega}(z^2) = \tilde{\Omega}(n^{1+\phi})$. Now, using the notation of Figure 7, we need to prove invariants (I) and (III), namely, that L (resp. H) contains all the light (resp. heavy) edges in $\mathcal{L}_t$.

Let $(x, y) \in \mathcal{L}_t$ be a light edge. By Lemma 7.5, the probability of collecting (x, y) at line 4a is $\Omega(z^{-2})$, thus $\tilde{O}(z^2)$ iterations are sufficient to collect (x, y) with high probability. Then, a union bound over all light edges shows that, with high probability, L contains all light edges in $\mathcal{L}_t$.

Since the $T = \tilde{O}(z^2)$ executions of line 4a are independent, standard concentration bounds ensure that for each 0.03-frequent vertex x, $c(x) \geq 0.02 \cdot T$ at line 8, and thus $x \in F$, with high probability. Likewise, with high probability, each vertex x that is not 0.01-frequent satisfies $c(x) < 0.02 \cdot T$ at line 8 and thus $x \notin F$. This proves invariant (II). Condition on invariant (II). By Lemma 7.6, we have that all heavy vertices are 0.03-frequent, and thus all heavy edges are incident to F. Since `FindClosePairs` runs an exhaustive search over the the sets $(X \cap F) \times Y$ and $X \times (Y \cap F)$, then invariant (III) must hold.

Finally, we bound the running time of `FindClosePairs`. We make $\tilde{O}(z^2)$ calls to `SubsCP`$(X, Y, 1/z)$. Thanks to Lemma 7.3, each of these calls takes time $\tilde{O}((n/z)^{2-\Omega(\phi)})$ with high probability. Thus, the combined time spent running `SubsCP` is

$$\tilde{O}(n^{1+\phi}) \cdot \left(n^{\frac{1}{2} - \frac{\phi}{2}}\right)^{2-\Omega(\phi)} = \tilde{O}(n^{1+\phi}) \cdot n^{1-(1+\Omega(1))\phi + \Theta(\phi^2)} = \tilde{O}(n^{2-\Omega(\phi)}).$$

The for loop at line 11 takes time $O(|F| \cdot n)$. Combining Lemma 7.7 and invariant (II), we obtain $|F| = O(z)$, thus the for loop at line 11 takes $O(n^{3/2+\phi/2}) = \tilde{O}(n^{2-\Omega(\phi)})$ time. $\qquad\square$

7.2 From All Close Pairs Retrieval to Sampling from λ, for Fixed D

The following theorem, is going to be employed in the remainder of this section.

Theorem 7 (Theorem 4 in [BT24]). *Let μ be a distribution supported over a size-n set X. Let $w : X \to \mathbb{R}_+$ be a function such that $\Pr_{\boldsymbol{y} \sim \mu}[\boldsymbol{y} = x] \propto w(x)$. Suppose that we can sample $\boldsymbol{y} \sim \mu$ and observe $w(\boldsymbol{y})$. Then, there exists an algorithm that takes $O(\sqrt{n}/\varepsilon)$ samples and estimates $W := \sum_{x \in X} w(x)$ up to a multiplicative error $1 \pm \varepsilon$.*

Lemma 7.8 (Sampling from λ for Fixed D). *Fix any $\phi, \eta \in (0, 1/2)$, $n \geqslant 1$, and $X, Y \subseteq (\mathbb{R}^d, \ell_1)$. Let $S \subseteq X \times Y$ a set of size $n^{2-\rho \cdot \phi}$ for some constant $\rho \in (0, 1/2)$. Let $C = C(S) \in \mathbb{R}^{n \times n}$ be a rounded distance matrix defined as in Definition 6.1. Fix an arbitrary matrix $P \in \{\pm 1\}^{n \times n}$ and let $D \in \mathbb{R}^{n \times n}$ be a matrix of all equal entries. Define the distribution $\lambda = \lambda(\eta, C, D, P)$ as in Definition 6.7.*

Suppose that we are given $L_t(X, Y)$ such that $|\mathcal{L}_t(X, Y)| = \tilde{O}(n^{1+\phi})$ and $|L_{t+3}(X, Y)| = \tilde{\Omega}(n^{1+\phi})$. Furthermore, suppose that we are given S, and that S τ-shatters the partition $\{L_\ell\}_{\ell \geqslant 0}$ of $X \times Y$ (Definition 8.1), for $\tau = n^{1+\phi/2}$. Then, there exists an algorithm $\texttt{ConstantSampler}$ that, with probability at least $1 - 1/\mathrm{poly}(n)$, generates n samples from λ, in time $\tilde{O}(n^{2-\Omega(\phi)})$.

Proof. Let $\kappa = D_{i,j}$ be the fixed value of all entries of D. Then recall that $\lambda_{i,j,\sigma} \propto w_{i,j,\sigma}$, where

$$w_{i,j,\sigma} = \exp\left(\eta \cdot \kappa \cdot \sigma \cdot \frac{P_{ij}}{C_{ij}}\right).$$

Where recall that $C_{i,j} = (1 + \varepsilon)^{\psi_{i,j} - 2S_{i,j}}$ where $\psi_{i,j}$ is the integer satisfying $(1 + \varepsilon)^{\psi_{i,j}} \leqslant \|x_i - y_j\|_1 \leqslant (1 + \varepsilon)^{\psi_{i,j}+1}$. To sample (i, j, σ) with probability proportional to $w_{i,j,\sigma}$, it suffices to first sample (i, j) with probability proportional to

$$w_{i,j} = \exp\left(\eta \cdot |\kappa| \cdot \frac{1}{C_{ij}}\right).$$

And then sample $\sigma \in \{1, -1\}$ with probability exactly $\frac{w_{i,j,\sigma}}{2 \cdot w_{i,j}}$, and reject if neither $\sigma = 1$ or $\sigma = -1$ is sampled. If sampled, it is clear that the resulting (i, j, σ) is drawn from λ. Finally, note that for exactly one value of $\sigma \in \{-1, 1\}$ we have $w_{i,j} = w_{i,j,\sigma}$, and therefore the probability that a sample is rejected is at most $1/2$. In what follows, let $\lambda_{i,j}$ be the distribution where $\mathbf{Pr}\left[(i, j)\right] \propto w_{i,j}$.

To sample proportionally to $w_{i,j}$, first, the algorithm exactly computes the values $w_{i,j}$ for all $(i, j) \in \mathcal{T} := \mathcal{L}_t \cup (S \cap (L_{t+1} \cup L_{t+2}))$, and the normalization factor $w_\mathcal{T} = \sum_{(i,j) \in \mathcal{T}} w_{i,j}$. For a set $T \subset [n] \times [n]$, let $\overline{T}$ be its complement, and let $\lambda|_T$ denote the distribution of λ conditioned on T. We show how to sample from both $\lambda|_\mathcal{T}$ and $\lambda|_{\overline{\mathcal{T}}}$.

First, for any $(i, j) \in \overline{\mathcal{T}}$, we know that $C_{i,j} \geqslant (1 + \varepsilon)^t$, since (1) we know $(i, j) \in L_{t'}$ for some $t' > t$ and (2) $(i, j) \notin S$ unless $t' \geqslant t + 3$ by definition. Set $w_{\max} = \exp(\eta \cdot |\kappa| \cdot (1 + \varepsilon)^{-t})$. Then $w_{i,j} \leqslant w_{\max}$ for all $(i, j) \in \overline{\mathcal{T}}$. Thus, to sample from $\lambda|_{\overline{\mathcal{T}}}$, we can perform the following steps: **(1)** sample $(i, j) \sim [n] \times [n]$ uniformly, **(2)** reject (i, j) if $(i, j) \notin \overline{\mathcal{T}}$, **(3)** keep the sample with probability $\frac{w_{i,j}}{w_{\max}}$, otherwise reject the sample. Clearly the distribution is correct, thus it remains to analyze the failure probability. First note that $|\overline{\mathcal{T}}| \geqslant n^2 - O(n^{2-\rho \cdot \phi} + n^{1+\phi}) \geqslant n^2/2$, thus the probability of rejection in step **(2)** is at most $1/2$. Next, note that by definition of τ-shattering, noting that $|L_{t+3}| \geqslant \tau$, we have that

$$\frac{|S \cap L_{t+3}|}{|S|} = \left(1 \pm \frac{1}{10}\right)\frac{|L_{t+3}|}{n^2} = \tilde{\Omega}\left(n^{-1+\phi}\right)$$

Thus $|S \cap L_{t+3}| \geqslant n^{1+\phi/2}$, and note that $w_{i,j} = w_{\max}$ for all $(i, j) \in S \cap L_{t+3}$. Thus, whenever (i, j) is sampled from $S \cap L_{t+3}$, we do not reject in step **(3)** in the above sampling procedure. Moreover, we sample such an (i, j) on step **(1)** with probability at least $\frac{|S \cap L_{t+3}|}{n^2} \geqslant \frac{1}{n^{1-\phi/2}}$. By Chernoff bounds (applied to the event that we successfully sample a (i, j) or reject it), with probability $1 - 1/\mathrm{poly}(n)$ it follows that we can compute n samples from $\lambda|_{\overline{\mathcal{T}}}$ in time $\tilde{O}(n^{2-\Omega(\phi)})$, as needed.

Finally, note that after $O(|S|)$ time pre-processing to compute $w_{i,j}$ for all $(i,j) \in \mathcal{T}$ as well as $w_{\mathcal{T}}$, we can sample from $\lambda|_{\mathcal{T}}$ in constant time. It remains to decide with what probability to sample from $\lambda|_{\mathcal{T}}$ versus $\lambda|_{\overline{\mathcal{T}}}$. Using Lemma Theorem 7, we can compute a value $\widehat{w_{\overline{\mathcal{T}}}}$ such that $\widehat{w_{\overline{\mathcal{T}}}} = (1 \pm 1/2)w_{\overline{\mathcal{T}}}$ where $w_{\overline{\mathcal{T}}} = \sum_{(i,j)\in\overline{\mathcal{T}}} w_{i,j}$. To sample from λ, we first chose to sample from $\mathcal{T}$ or $\overline{\mathcal{T}}$ with probability $\frac{w_{\mathcal{T}}}{w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}}}$ or $\frac{\widehat{w_{\overline{\mathcal{T}}}}}{w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}}}$ respectively. Conditioned on the procedure not rejecting a sample, our procedure samples a pair (i,j) with probability $\widehat{\lambda_{i,j}}$ where $\widehat{\lambda_{i,j}} = (1 \pm 1/2)\lambda_{i,j}$, since the only source of error is in our approximation $\widehat{w_{\overline{\mathcal{T}}}}$ of $w_{\overline{\mathcal{T}}}$.

We now show how to sample exactly from the distribution $\lambda_{i,j}$. To see this, note that whenever we sample a pair (i,j) from $\widehat{\lambda}$, we can compute exactly the probability $\widehat{\lambda_{i,j}}$ with which we sampled it; this is clear in the case $(i,j) \in \mathcal{T}$, and otherwise the probability is precisely $\frac{1}{|\overline{\mathcal{T}}|} \cdot \frac{w_{i,j}}{w_{\max}}$, and $|\overline{\mathcal{T}}| = n^2 - |\mathcal{T}|$ can be computed exactly since $\mathcal{T}$ is computed exactly. Thus, to sample exactly from λ, we first sample (i,j) from $\widehat{\lambda}$, and reject it with probability $1 - \frac{1}{\widehat{\lambda_{i,j}}} \cdot \frac{w_{i,j}}{4(w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}})}$. First note that this is a valid probability, because $\frac{w_{i,j}}{4(w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}})} < \lambda_{i,j}/2 \leqslant \widehat{\lambda_{i,j}}$. Overall, the probability we sample and keep (i,j) is

$$\widehat{\lambda_{i,j}} \cdot \frac{1}{\widehat{\lambda_{i,j}}} \cdot \frac{w_{i,j}}{4(w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}})} = \frac{w_{i,j}}{4(w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}})}$$

which is proportional to $w_{i,j}$. Thus conditioned on keeping the pair, we sample from precisely the correct distribution. Finally, note that $\frac{w_{i,j}}{4(w_{\mathcal{T}}+\widehat{w_{\overline{\mathcal{T}}}})} > \frac{1}{8}\lambda_{i,j} > \frac{1}{16}\widehat{\lambda_{i,j}}$, thus we reject with probability at most $15/16$, which can be corrected for by oversampling by a $O(1)$ factor, completing the proof. $\qquad\square$

7.3 From Fixed D to Arbitrary D

Lemma 7.9. *Fix any $\eta, \varepsilon, \phi \in (0, 1/2)$, $\chi \geqslant 1/\mathrm{poly}(\varepsilon^{-1}, \log(n\cdot\Phi))$, $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{Z}^{2n}$ with $||\boldsymbol{\alpha}||_\infty, ||\boldsymbol{\beta}||_\infty \leqslant \mathrm{poly}(n\Phi)$, and, $C \in \mathbb{R}^{n\times n}$ be a distance matrix. Define $P \in \{\pm 1\}^{n\times n}$ and $D \in \mathbb{R}^{n\times n}$ as in Definition 6.6. Define $\lambda \in \mathcal{D}(\eta, C, D, P)$ as in Definition 6.7.*

Suppose that there exists a randomized algorithm `ConstantSampler` that returns n samples from λ and runs in time $\tilde{O}(n^{2-\Omega(\phi)})$, assuming that D has all-equal entries. Then, for each constant $z > 0$, there exists an algorithm `ArbitrarySampler` that returns $n \cdot \log^z(n)$ samples from a distribution λ', with $|\lambda' - \lambda|_{\mathrm{TV}} = o\left(\frac{\varepsilon^2}{\log n \cdot \log^2 \Phi}\right)$ and runs in time $\tilde{O}(n^{2-\Omega(\phi)} \cdot \mathrm{poly}(\varepsilon^{-1}))$, for any D defined in Definition 6.6.

Moreover, `ArbitrarySampler` calls `ConstantSampler` on sets $\{X_i \times Y_i\}_{i\in[k]}$, were $k = O(n^{3/2})$, $|X_i| = |Y_i| = \sqrt[4]{n}$ and $\{X_i \times Y_i\}_{i\in[k]}$ are pairwise disjoint. If `ConstantSampler` on is correct on all such calls, then `ArbitrarySampler` is correct, with high probability.

Proof. Our proof strategy is as follows: we partition the triples i, j, σ into groups where the entries of D are constant, then first sample one of the groups and finally resort to `ConstantSampler` to sample a pair within the group.

Definition 6.6 and $||\boldsymbol{\alpha}||_\infty, ||\boldsymbol{\beta}||_\infty \leqslant \mathrm{poly}(n\Phi)$ imply that the entries of D belong to $\{0\} \cup \{(1+\chi)^s : s \in \mathbb{Z} \cap [-T, T]\}$ for $T = O(\chi^{-1} \cdot \log(n\Phi))$. Define $Q_s := \{(i,j) \in [n]^2 \mid D_{ij} = (1+\chi)^s\}$, $Q_\circ := \{(i,j) \in [n]^2 \mid D_{ij} = 0\}$ and notice that $Q_\circ \cup \bigcup_{s\in[-T,T]} Q_s$ is a partition of $[n]^2$. Our strategy is to construct a partition of $[n] \times [n]$ given by $E \cup \bigcup_{\ell\in[L]} I_\ell \times J_\ell$ satisfiying the following desiderata:

(i) $|I_\ell| = |J_\ell| = \sqrt[4]{n}$,

(ii) $I_\ell \times J_\ell \subseteq Q_s$ for some $s \in \{\circ\} \cup [-T, T]$.

(iii) $|E| = O(n^{7/4})$.

We construct such partition processing one Q_s at a time. Then, our sampling strategy will first sample a set $S \in \{E\} \cup \{I_\ell \times J_\ell\}_{\ell \in [L]}$ and then sample a triplet $(i, j, \sigma) \in S \times \{\pm 1\}$ via `ConstantSampler`.

Computing our partition. Here we describe the algorithm that constructs the desired partition. Set $\ell \leftarrow 0$ and $E \leftarrow \varnothing$. We start from $Q_\circ$. Given v, it is straightforward to construct an implicit representation of the equivalence relation $i \sim j \iff \alpha_i = \beta_j$ in near-linear time by sorting α and β. For each $x \in \{\alpha_i\}_{i \in [n]}$ if the equivalence class of x, call it $C_x := \{j \in [n] \mid \beta_j = x\}$, has size at least $\sqrt{n}$, then we define $I_\ell = J_\ell = C_x$ and update $\ell \leftarrow \ell + 1$. Else, if $|C_x| \leqslant \sqrt{n}$, then we update $E \leftarrow E \cup C_x \times C_x$.

For each $s \in [-T, T]$ we do the following. Sort α_i so that $i \mapsto \alpha_i$ is increasing, and do the same for β_j. For all $i \in [n]$, the set $Q_s \cap \{i\} \times [n]$ is an interval and we denote it with $[a(i), b(i)]$. Let $q_j = j \cdot \sqrt{n}$ for $j \in [\sqrt{n}]$. Then, we execute the loop in Figure 8.

For $j \in [\sqrt{n}]$ do:

1. $E \leftarrow E \cup ([q_j, q_{j+1}) \times [a(q_j), a(q_{j+1})) \cap Q_s)$.

2. $E \leftarrow E \cup ([q_j, q_{j+1}) \times (b(q_j), b(q_{j+1})] \cap Q_s)$.

3. If $a(q_{j+1}) + \sqrt{n} \leqslant b(q_j)$, then $I_\ell \leftarrow [q_j, q_{j+1})$, $J_\ell \leftarrow [a(q_{j+1}), b(q_j)]$ and $\ell \leftarrow \ell + 1$.

4. Else, $E \leftarrow E \cup ([q_j, q_{j+1}) \times [a(q_{j+1}), b(q_j)])$.

Figure 8: Partitioning Q_s .

We start by proving that our algorithm outputs a partition. First, we observe that for each $s \in \{\circ\} \cup [-T, T]$ our algorithm partitions the elements of Q_s between E and several sets $I_\ell \times J_\ell$. Indeed, for $s = 0$ this is trivial. For $s \in [-T, T]$, we notice that all elements of Q_s are included because $a(q_j) \leqslant a(q_{j+1}), b(q_j) \leqslant b(q_{j+1})$ and $Q_s \cap [q_j, q_{j+1}) \times [n] \subseteq [q_j, q_{j+1}) \times [a(q_j), b(q_{j+1})]$. Moreover, no element is included twice because all intervals are disjoint by definition and $[q_j, q_{j+1}) \times [a(q_{j+1}), b(q_j)] \subseteq Q_s$.

Next, we prove that the running time does not exceed $O(n^{3/2} \cdot T)$. For $s = \circ$ the running time is $\tilde{O}(n)$ as sorting suffices. For generic $s \in [-T, T]$ we prove that our algorithm runs in time $O(n^{3/2})$. For each i, computing $a(i)$ and $b(i)$ can be done in $\log n$ time through binary search. Computing the intersection $[q_j, q_{j+1}) \times [a(q_j), a(q_{j+1})) \cap Q_s$ can be done in time $\sqrt{n} \cdot |a(q_j) - a(q_{j+1})|$, which summing over j gives $\sqrt{n} \cdot \sum_{j \in [\sqrt{n}]} |a(q_j) - a(q_{j+1})| = n^{3/2}$. Likewise, we can bound the total time used to compute the intersection $[q_j, q_{j+1}) \times (b(q_j), b(q_{j+1})) \cap Q_s$.

Now, we prove that our partition satisfies the desiderata. First, we prove that $|E| = O(n^{7/4})$. For $s = \circ$, we have that all sets added to E are of the form $C_x \times C_x$ for $|C_x| \leqslant \sqrt{n}$. Since all C_x are disjoint, we have $n^{3/2} \geqslant |E \cap Q_0|$. For generic $s \in [-T, T]$, the proof follows the same computation

that we performed to bound the running time. In steps (1) and (2) we increase $|E|$ by at most $\sqrt{n} \cdot |a(q_j) - a(q_{j+1})|$ and $\sqrt{n} \cdot |b(q_j) - b(q_{j+1})|$ respectively, and summing over j, we increase $|E|$ by at most $O(n^{3/2})$. In step (4), we are promised that $b(q_j) - a(q_{j+1}) < \sqrt{n}$, so we add at most $|q_{j+1} - q_j| \cdot \sqrt{n} = n$ elements to E. Summing over all $j \in [\sqrt{n}]$ we obtain $n^{3/2}$.

Next, we refine the sets I_ℓ and J_ℓ into smaller sets to make sure that each smaller set of pairs that we create has size $\sqrt[4]{n} \times \sqrt[4]{n}$. Notice that we have $|I_\ell|, |J_\ell| \geqslant \sqrt{n}$ for each ℓ. Indeed, the above holds trivially for $s = \circ$, and for generic $s \in [-T, T]$ it holds because of the guard condition at step (3). Then, if $\sqrt[4]{n}$ divides $|J_\ell|$ it suffices to split I_ℓ (resp. J_ℓ) into size-$\sqrt[4]{n}$ chunks $I_{\ell,1} \ldots I_{\ell,t}$ (resp. $J_{\ell,1} \ldots J_{\ell,t'}$) and consider all pairwise products $I_{\ell,a} \times J_{\ell,b}$ for $a \in [t]$, $b \in [t']$. If $\sqrt[4]{n}$ does *not* divide J_ℓ, then we have one leftover chunk $\tilde{J}_\ell$ of size at most $\sqrt[4]{n}$, an we add $I_\ell \times \tilde{J}_\ell$ to E. Since $|\tilde{J}_\ell|/|J_\ell| \leqslant 1/\sqrt[4]{n}$, the size of E increases by at most $n^{7/4}$. Finally, desiderata (iii) is clearly satisfied since we construct $I_\ell \times J_\ell$ as a refinement of some Q_s. Thus, we satisfy all desiderata (i)–(iii).

Two-step sampling. Fix $\delta = \frac{\varepsilon^2}{\log^2 n \cdot \log^2 \Phi}$. Recall how weights are defined in Definition 6.7:

$$w_{i,j,\sigma} = \exp(\eta \cdot (\sigma \cdot P_{ij}) \cdot D_{ij}/C_{ij}),$$

so, whenever we sample (i, j, σ) we can compute its weight, as we have access to C and, thorough $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, also to P and D. Thus, for each set $S \in \{I_\ell \times J_\ell\}_{\ell \in [L]}$, one can compute an approximation $\tilde{v}(S)$ of the volume $v(S) := \sum_{(i,j) \in S, \sigma \in \{\pm 1\}} w_{i,j,\sigma}$ using Theorem 7 so that $\tilde{v}(S) = v(S)(1 \pm \delta)$ in time $O(\sqrt[4]{n}/\delta) = \tilde{O}(\sqrt[4]{n} \cdot \text{poly}(\varepsilon^{-1}))$. Since $L \leqslant n^2/(\sqrt[4]{n} \cdot \sqrt[4]{n}) = n^{3/2}$, this can be done in total time $\tilde{O}(n^{7/4} \cdot \text{poly}(\varepsilon^{-1}))$. As for E, we can compute $v(E)$ exactly in time $|E| = O(n^{7/4})$.

Our final sampling procedure first sample a part S of the partition proportionally to $\tilde{v}(S)/\sum_{S'} \tilde{v}(S')$ and then uses `ConstantSampler` to sample a $(i, j, \sigma) \in S$. The TV distance between this sampling distribution and the distribution λ only comes from the approximation of $v(S)$. Observe that, assuming correctness of `ConstantSampler`, the total variation distance between the sampling distribution induced by the $v(S)$ and the distribution induced by the sampling with the $\tilde{v}(S)$ is at most $O(\delta) = o\left(\frac{\varepsilon^2}{\log n \cdot \log^2 \Phi}\right)$.

Each call to `ConstantSampler` on $I_\ell \times J_\ell$ returns $\sqrt[4]{n}$ samples, so, in order to collect $n \cdot \log^z(n)$ samples, `ArbitrarySampler` needs to call `ConstantSampler` at most $L + n \cdot \log^z(n)/\sqrt[4]{n} = \tilde{O}(n^{3/2})$ times.

The running time of the sampling phase has two terms. The first term comes from approximating the volume of each partition, and it is $\tilde{O}(n^{7/4} \cdot \text{poly}(\varepsilon^{-1}))$. The second term comes from running `ConstantSampler`. As noted above, the number of such calls is bounded by $\tilde{O}(n^{3/2})$ and each calls uses time $\tilde{O}((\sqrt[4]{n})^{2-\Omega(\phi)})$, so, the total time spent running `ConstantSampler` is $\tilde{O}(n^{2-\Omega(\phi)} \cdot \text{poly}(\varepsilon^{-1}))$. $\qquad \square$

8 Designing a Consistent Rounding

Definition 8.1. *Given a partition $A = A_1 \cup \cdots \cup A_k$, we say that $D \subseteq A$ is τ-shatters the partition, for some $\tau \geqslant 1$, if for all i such that $|A_i| \geqslant \tau$, we have*

$$0.9 \cdot \frac{|A_i|}{|A|} \leqslant \frac{|D \cap A_i|}{|D|} \leqslant 1.1 \cdot \frac{|A_i|}{|A|}.$$

For two sets A, B, let $\mathcal{F}_{A,B} = \{f : A \to B\}$ be the set of all functions from A to B. Also for a set A, let $A[t] = \{S \subseteq A \mid |S| \leqslant t\}$.

Lemma 8.2. *Fix any $\phi \in (0,1)$, $k, n \geqslant 1$, let A be a set of size n, Σ be an arbitrary set, $\mathsf{R} = \mathrm{polylog}(n)$, and $t = \tilde{O}(\sqrt{n})$. For $i \in [\mathsf{R}]$, consider any fixed sequence of functions $(h_i, g_i, w_i)_{i \in [\mathsf{R}]}$ where $h_i : \Sigma \times A[t] \to \Sigma$, $g_i : \Sigma \times A[n] \to A[t]$, and $w_i : \Sigma \to \mathcal{F}_{A,[k]}$. Fix any $\sigma_1 \in \Sigma$. Suppose $D \subseteq A$ is a set of $m = \Theta(n^{1-\phi/2})$ uniformly sampled points and consider the sequences $\sigma_1 \ldots \sigma_\mathsf{R}$ and $f_1 \ldots f_\mathsf{R}$ given by*

$$f_i = w_i(\sigma_i) \quad \text{and} \quad \sigma_{i+1} = h_i(\sigma_i, g_i(\sigma_i, D)).$$

Define the partitions of A given by $A_i^{(j)} = f_i^{-1}(j)$ for each $i \in [\mathsf{R}]$. Then, with probability $1 - \exp(-\Omega(\sqrt{n}))$, for each $i \in [\mathsf{R}]$, the set D τ-shatters the partition $A^{(i)}$ for $\tau = n^{1/2+\phi}$.

Proof. First note that any function $f \in \mathcal{F}_{A,k}$ gives rise to a natural partition of A via $\{f^{-1}(j)\}_{j \in [k]}$, thus in what follows refer this as the partition corresponding to f. The critical observation is that the value of $f_{i+1} = w_i \circ h_i \circ g_i(\sigma_i, D)$ depends only on a set $S_i = g_i(\sigma_i, D)$ of at most t elements from A. Thus, there are only $\binom{n}{t} \leqslant n^t = 2^{\tilde{O}(\sqrt{n})}$ possible values for the set S_i. It follows that there are at most $2^{\tilde{O}(\sqrt{n})}$ partitions which can arise from the function f_i, for any $i \in [\mathsf{R}]$, and therefore a total of $n^{tz \cdot \mathsf{R}} = 2^{\tilde{O}(\sqrt{n})}$ sequences of partitions which can arise overall by the functions $f_1, \ldots, f_\mathsf{R}$, which are moreover fixed in advance and independent of D. Denote by $\mathcal{P}$ the set of all such sequences of partitions.

Now for any fixed partition $A = A_1 \cup \cdots \cup A_k$, and fix any i such that $|A_i| \geqslant \tau = n^{1/2+\phi}$ (if one exists), and let $X_j \in \{0,1\}$ indicate that the j-th sample in D is contained in A_i. Clearly $\mathbb{E}[X_i] = \frac{|A_i|}{|A|}$. Thus, by Chernoff bounds, we have

$$\mathbf{Pr}\left[\left|\sum_{i=1}^{m} X_i - \frac{m|A_i|}{|A|}\right| \geqslant \frac{1}{20}\frac{m|A_i|}{|A|}\right] \leqslant \exp(-\frac{1}{2000}\frac{m|A_i|}{|A|}) \tag{17}$$
$$\leqslant \exp(-\frac{1}{2000}n^{1/2+\phi/2})$$

Conditioned on this event that $\left|\sum_{i=1}^{m} X_i - \frac{m|A_i|}{|A|}\right| \leqslant \frac{1}{20}\frac{m|A_i|}{|A|}$, we have

$$\frac{|D \cap A_i||}{|A_i|} = \frac{|D|}{|A|}\left(1 \pm \frac{1}{20}\right)$$

Thus, by a union bound over all $i \in [k]$, it follows that the sample D τ-shatters to the partition $A_1, \ldots, A_k$ with probability at least $k \exp(-\frac{1}{2000}n^{1/2+\phi/2})$. We then have

$$\mathbf{Pr}\left[D \text{ } \tau\text{-shatters all partitions in all sequences in } \mathcal{P}\right] \geqslant 1 - 2^{\tilde{O}(\sqrt{n})} \cdot \exp\left(-\frac{1}{2000}n^{1/2+\phi/2}\right) \tag{18}$$
$$\geqslant 1 - \exp\left(-\frac{1}{4000}n^{1/2+\phi/2}\right)$$

which completes the proof $\qquad\square$

Corollary 8.3. *Fix any $\phi \in (0,1)$ and $k, n \geqslant 1$. Let A be a set of size n^2 and Σ be an arbitrary set. Pick $D \subseteq A$ of size $n^{2-\phi/8}$ uniformly at random and let $\mathsf{R} = polylog(n)$. For $i \in [\mathsf{R}]$, let p_i be any probability distribution over A. For $i \in [\mathsf{R}]$, let $S_i \subseteq A$ the the multi-set of $\tilde{O}(n)$ i.i.d. samples from p_i.*

Consider sequences of "state" variables $\sigma_1 \ldots \sigma_\mathsf{R} \in \Sigma$ and "paritioning functions" $f_1 \ldots f_\mathsf{R} \in \mathcal{F}_{A,[k]}$. Fix $\sigma_1 \in \Sigma$. Suppose that, for $i \in [\mathsf{R}-1]$: (i) σ_{i+1} depends on the sample S_i and on σ_i; (ii) p_i depends on σ_i and D; (iii) f_i depends on σ_i. Define the partitions of A given by $A_i^{(j)} = f_i^{-1}(j)$ for each $i \in [\mathsf{R}]$. Then, with high probability, for each $i \in [\mathsf{R}]$, D τ-shatters the partition $\{A_i^{(j)}\}_{j \in B}$ with $\tau = n^{1+\phi/2}$.

Proof. Follows immediately from Lemma 8.2, where g_i is the process which samples the set S_i given the fixed value of D and the current "state" σ_i; h_i is the function which generates σ_{i+1} from the samples S_i and σ_i; and w_i is the function that defines the partitioning function f_i given the state σ_i. Note that these functions are randomized, but Lemma 8.2 holds for any possible fixing of the randomness, as it holds for any functions g_i, h_i, w_i. $\qquad\square$

A Reduction to Polynomial Aspect Ratio

Lemma A.1. *There exists a randomized algorithm which runs in $O(nd + n \log n)$ time and has the following guarantees:*

- **Input**: *A set $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$ and a vector $b \in V$.*

- **Output**: *A random variable $\boldsymbol{\eta} \in \mathbb{R}_{\geqslant 0}$.*

With probability $1 - o(1)$ over the randomness of the algorithm,

$$\mathsf{EMD}_X(b) \leqslant \boldsymbol{\eta} \leqslant \tilde{O}(n^2 \sqrt{d}) \cdot \mathsf{EMD}_X(b).$$

Proof. The algorithm proceeds by reducing to a one-dimensional problem in the following way:

1. Sample a random vector $\boldsymbol{g} \sim \mathcal{N}(0, I_d)$ and let $\mathbf{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\} \subset \mathbb{R}$ be given by $\boldsymbol{y}_i = C \cdot \langle x_i, \boldsymbol{g} \rangle$, for a parameter $C = O(n^2 \sqrt{d})$. Sort and re-index points so $\boldsymbol{y}_1 \leqslant \cdots \leqslant \boldsymbol{y}_n$. Note that this transformation takes $O(nd) + O(n \log n)$ to project and sort.

2. Compute $\mathsf{EMD}_{\mathbf{Y}}(b)$ in $O(n)$ time via a "two-pointer" algorithm. This is possible because the optimal coupling in one-dimension proceeds by greedily mapping as much supply/demand in sorted order, so one may proceed from smallest-to-largest by maintaining one pointer to current excess supply, and one to current excess demand.

It remains to prove the approximation guarantees, which follow from the fact that the map $(\mathbb{R}^d, \ell_1) \to (\mathbb{R}^1, \ell_2)$ given by the scaled projection onto $\boldsymbol{g}$ is $\tilde{O}(n^2 \sqrt{d})$-bi-Lipschitz with probability $1 - o(1)$. This follows from the fact the identity $\ell_1^d \to \ell_2^d$ can only contract distances by at most $\sqrt{d}$, and the fact that with probability $1 - o(1)$ over $\boldsymbol{g}$, $1/(n^2 \log n) \cdot \|x_i - x_j\|_2 \leqslant \langle \boldsymbol{g}, x_i - x_j \rangle \leqslant O(\sqrt{\log n}) \|x_i - x_j\|_2$. Since we compute $\mathsf{EMD}_{\mathbf{Y}}(b)$ optimally, we obtain the desired guarantees on $\mathsf{EMD}_X(b)$. $\qquad\square$

Lemma A.2. *There is a randomized algorithm with the following guarantees:*

- **Input**: *A set $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$, and a vector $b \in V$ with integer coordinates.*

- **Output**: *A partition $X_1, \ldots, X_t$ of X which induces a partition of the vector b into $b_1, \ldots, b_t$.*

With probability 0.9, we have that each $b_1, \ldots, b_t \in V$, and that

$$\mathsf{EMD}_X(b) = \sum_{i=1}^{t} \mathsf{EMD}_{X_i}(b_i).$$

Furthermore, the maximum pairwise distance in X_i is at most $\mathrm{poly}(nd) \cdot \mathsf{EMD}_X(b)$.

Proof. The algorithm proceeds by first executing the algorithm of Lemma A.1 in order to obtain an approximation of $\mathsf{EMD}_X(b)$ in $\boldsymbol{\eta}$. We assume the algorithm succeeds (which occurs with probability at least $1 - o(1)$) so $\boldsymbol{\eta}$ is at least $\mathsf{EMD}_X(b)$ and at most $\tilde{O}(n^2\sqrt{d}) \cdot \mathsf{EMD}_X(b)$. The partition of X proceeds by imposing a randomly shifted grid of side-length $100\boldsymbol{\eta}$, since this means that any pair x_i, x_j fall in different parts with probability at most $\|x_i - x_j\|_1/(100\boldsymbol{\eta})$. In particular, we sample a random vector $\boldsymbol{h} \sim [0, 100\boldsymbol{\eta}]^d$ and we let $X_1, \ldots, X_t$ be the sets consisting of non-empty parts for each integer $a_1, \ldots, a_d$,

$$\{x_i \in X : \boldsymbol{h} + a_\ell \cdot \boldsymbol{\eta} \leqslant x_{i\ell} < \boldsymbol{h} + (a_\ell + 1) \cdot \boldsymbol{\eta}\}.$$

Since $b \in V$ is an integer vector, the integrality of min-cost flows implies that there exists a minimizing feasible flow $\gamma \in \mathbb{R}_{\geqslant 0}^{n \times n}$ with integer entries. We let

$$\mathbf{S} = \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \cdot \mathbf{1}\{x_i, x_j \text{ fall in different parts}\},$$

and note that the randomly shifted grid construction implies

$$\mathbf{E}[\mathbf{S}] = \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} \cdot \frac{\|x_i - x_j\|_1}{100 \cdot \boldsymbol{\eta}} \leqslant \frac{\mathsf{EMD}_X(b)}{100 \cdot \boldsymbol{\eta}} \leqslant \frac{1}{100}.$$

Since $\mathbf{S}$ is a non-negative integer, it must be 0 with probability at least $99/100$. In this case, in an optimal coupling, no coupled pairs fall in different parts. This implies $b_1, \ldots, b_t \in V$, and that we may compute $\mathsf{EMD}_X(b)$ by summing $\mathsf{EMD}_{X_i}(b_i)$. The final claim of maximum pairwise distance follows from the diameter of each grid cell $O(d) \cdot \boldsymbol{\eta}$, which is at most $\mathrm{poly}(nd) \cdot \mathsf{EMD}_X(b)$. $\qquad\square$

The above lemma upper bounds the maximum distance, and the following claim applies a small amount of noise in order to lower bound the minimum distance while ensuring that the Earth mover's distance does not change significantly.

Claim A.3. *For any $X = \{x_1, \ldots, x_n\} \subset (\mathbb{R}^d, \ell_1)$ and $\varepsilon > 0$, let $\mathbf{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\} \subset (\mathbb{R}^{d+d'}, \ell_1)$ with $d' = \Theta(\log n)$ where*

$$\boldsymbol{y}_i = (x_i, \boldsymbol{z}_i) \in \mathbb{R}^{d+d'},$$

with $\boldsymbol{z}_i \sim [0, \varepsilon]^{d'}$. Then, the minimum distance $\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_1 = \Omega(\varepsilon d')$ with probability $1 - o(1)$, and for any $b \in V$,

$$|\mathsf{EMD}_X(b) - \mathsf{EMD}_\mathbf{Y}(b)| \leqslant O(\varepsilon n \log n).$$

References

[ACW16a]   Josh Alman, Timothy M Chan, and Ryan Williams. "Polynomial representations of threshold functions and algorithmic applications". In: *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2016, pp. 467–476.

[ACW16b]   Josh Alman, Timothy M. Chan, and Ryan Williams. "Polynomial Representations of Threshold Functions with Applications". In: *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2016)*. To appear. 2016.

[Aga+17]   Pankaj Agarwal, Kyle Fox, Debmalya Panigrahi, Kasturi Varadarajan, and Allen Xiao. "Faster algorithms for the geometric transportation problem". In: *Proceedings of the 33rd International Symposium on Computational Geometry (SOCG '2017)*. 2017.

[Aga+22]   Pankaj K Agarwal, Hsien-Chih Chang, Sharath Raghvendra, and Allen Xiao. "Deterministic, near-linear ε-approximation algorithm for geometric bipartite matching". In: *Proceedings of the 54nd ACM Symposium on the Theory of Computing (STOC '2022)*. 2022.

[Aga+90]   Pankaj K Agarwal, Herbert Edelsbrunner, Otfried Schwarzkopf, and Emo Welzl. "Euclidean minimum spanning trees and bichromatic closest pairs". In: *Proceedings of the sixth annual symposium on Computational geometry*. 1990, pp. 203–210.

[AI08]   Alexandr Andoni and Piotr Indyk. "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions". In: *Communications of the ACM* 51.1 (2008), pp. 117–122.

[AIK08a]   Alexandr Andoni, Piotr Indyk, and Robert Krauthgamer. "Earth mover distance over high-dimensional spaces". In: *Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms (SODA '2008)*. 2008, pp. 343–352.

[AIK08b]   Alexandr Andoni, Piotr Indyk, and Robert Krauthgamer. "Earth mover distance over high-dimensional spaces." In: *SODA*. Vol. 8. 2008, pp. 343–352.

[Alt+93]   Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. "On sparse spanners of weighted graphs". In: *Discrete & Computational Geometry* 9.1 (1993), pp. 81–100.

[And+14]   Alexandr Andoni, Aleksandar Nikolov, Krzysztof Onak, and Grigory Yaroslavtsev. "Parallel algorithms for geometric graph problems". In: *Proceedings of the 46th ACM Symposium on the Theory of Computing (STOC '2014)*. 2014.

[And09]   Alexandr Andoni. "Nearest Neighbor Search: the Old, the New, and the Impossible". PhD thesis. MIT, 2009.

[AR15]   Alexandr Andoni and Ilya Razenshteyn. "Optimal Data-Dependent Hashing for Approximate Near Neighbors". In: *Proceedings of the 47th ACM Symposium on the Theory of Computing (STOC '2015)*. Available as arXiv:1501.01062. 2015, pp. 793–801.

[AS14a]   Pankaj K Agarwal and R Sharathkumar. "Approximation algorithms for bipartite matching with metric and geometric costs". In: *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 2014, pp. 555–564.

[AS14b]   Pankaj K. Agarwal and R. Sharathkumar. "Approximation algorithms for bipartite matching with metric and geometric costs". In: *Proceedings of the 46th ACM Symposium on the Theory of Computing (STOC '2014)*. 2014, pp. 555–564.

[ASZ20]   Alexandr Andoni, Clifford Stein, and Peilin Zhong. "Parallel approximate undirected shortest paths via low hop emulators". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2020. Chicago, IL, USA: Association for Computing Machinery, 2020, pp. 322–335. ISBN: 9781450369794. DOI: 10.1145/3357713.3384321. URL: https://doi.org/10.1145/3357713.3384321.

[AZ23]   Alexandr Andoni and Hengjie Zhang. "Sub-quadratic $(1+\varepsilon)$-approximate Euclidean Spanners, with Applications". In: *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2023, pp. 98–112.

[BR24]   Lorenzo Beretta and Aviad Rubinstein. "Approximate Earth Mover's Distance in Truly-Subquadratic Time". In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 2024, pp. 47–58.

[BT24]   Lorenzo Beretta and Jakub Tětek. "Better sum estimation via weighted sampling". In: *ACM Transactions on Algorithms* 20.3 (2024), pp. 1–33.

[Cha+23]   Moses Charikar, Beidi Chen, Christopher Ré, and Erik Waingarten. "Fast Algorithms for a New Relaxation of Optimal Transport". In: *Proceedings of the 36rd Annual Conference on Learning Theory (COLT '2023)*. 2023.

[Cha02]   Moses Charikar. "Similarity estimation techniques from rounding algorithms". In: *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC '2002)*. 2002, pp. 380–388.

[Che+22a]   Li Chen, Rasmus Kyng, Yang P Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. "Maximum flow and minimum-cost flow in almost-linear time". In: *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 612–623.

[Che+22b]   Xi Chen, Rajesh Jayaram, Amit Levi, and Erik Waingarten. "New streaming algorithms for high dimensional EMD and MST". In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. 2022, pp. 222–233.

[Cla87]   Ken Clarkson. "Approximation algorithms for shortest path motion planning". In: *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 1987, pp. 56–65.

[FL23]   Emily Fox and Jiashuai Lu. "A deterministic near-approximation scheme for geometric transportation". In: *Proceedings of the 64th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2023')*. 2023.

[HIM12]   Sariel Har-Peled, Piotr Indyk, and Rajeev Motwani. "Approximate Nearest Neighbor: Towards Removing the Curse of Dimensionality". In: *Theory of Computing* 8.1 (2012), pp. 321–350.

[HIS13]   Sariel Har-Peled, Piotr Indyk, and Anastasios Sidiropoulos. "Euclidean spanners in high dimensions". In: *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA '2013)*. 2013.

[IM98]   Piotr Indyk and Rajeev Motwani. "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality". In: *Proceedings of the 30th ACM Symposium on the Theory of Computing (STOC '1998)*. 1998, pp. 604–613.

[Ind01]   Piotr Indyk. "High-Dimensional Computational Geometry". PhD thesis. Stanford University, 2001.

[IT03]   Piotr Indyk and Nitin Thaper. "Fast Color Image Retrieval via Embeddings". In: *Workshop on Statistical and Computational Theories of Vision (at ICCV)*. 2003.

[JS82]   William B Johnson and Gideon Schechtman. "Embedding lpm into l 1 n". In: *Acta Mathematica* 149 (1982), pp. 71–85.

[KNP19]   Andrey Boris Khesin, Aleksandar Nikolov, and Dmitry Paramonov. "Preconditioning for the geometric transportation problem". In: *Proceedings of the 35th International Symposium on Computational Geometry (SoCG '2019)*. 2019.

[Li20a]   Jason Li. "Faster parallel algorithm for approximate shortest path". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2020. Chicago, IL, USA: Association for Computing Machinery, 2020, pp. 308–321. ISBN: 9781450369794. DOI: 10.1145/3357713.3384268. URL: https://doi.org/10.1145/3357713.3384268.

[Li20b]   Jason Li. "Faster parallel algorithm for approximate shortest path". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. 2020, pp. 308–321.

[Li21]   Jason Li. "Preconditioning and Locality in Algorithm Design." PhD thesis. Carnegie Mellon University, USA, 2021.

[PC19]   Gabriel Peyré and Marco Cuturi. "Computational Optimal Transport: With Applications to Data Science". In: *Foundations and Trends® in Machine Learning* 11.5–6 (2019), pp. 355–607.

[PS89]   David Peleg and Alejandro A Schäffer. "Graph spanners". In: *Journal of graph theory* 13.1 (1989), pp. 99–116.

[Roh19]   Dhruv Rohatgi. "Conditional Hardness of Earth Mover Distance". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2019.

[RR98]   Y. Rabinovich and R. Raz. "Lower Bounds on the Distortion of Embedding Finite Metric Spaces in Graphs". In: *Discrete & Computational Geometry* 19.1 (1998), pp. 79–94. DOI: 10.1007/PL00009336. URL: https://doi.org/10.1007/PL00009336.

[RS91]   Jim Ruppert and Raimund Seidel. "Approximating the d-dimensional complete Euclidean graph". In: *Proceedings of the 3rd Canadian Conference on Computational Geometry (CCCG 1991)*. 1991, pp. 207–210.

[RT81]   Edward M Reingold and Robert E Tarjan. "On a greedy heuristic for complete matching". In: *SIAM Journal on Computing* 10.4 (1981), pp. 676–681.

[Rub18]   Aviad Rubinstein. "Hardness of approximate nearest neighbor search". In: *Proceedings of the 50th annual ACM SIGACT symposium on theory of computing*. 2018, pp. 1260–1268.

[SA12]   R Sharathkumar and Pankaj K Agarwal. "A near-linear time ε-approximation algorithm for geometric bipartite matching". In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 2012, pp. 385–394.

[SA20]   R. Sharathkumar and Pankaj K. Agarwal. "A near-ε-approximation algorithm for bipartite geometric matching". In: *Journal of the ACM* 67.3 (2020), 18:1–18:19.

[San15]   Filippo Santambrogio. *Optimal transport for applied mathematicians*. Vol. 87. Springer, 2015.

[She13]   Jonah Sherman. "Nearly maximum flows in nearly linear time". In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE. 2013, pp. 263–269.

[She17]   Jonah Sherman. "Generalized preconditioning and undirected minimum cost flow". In: *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA '2017)*. 2017.

[Val15]   Gregory Valiant. "Finding Correlations in Subquadratic Time, with Applications to Learning Parities and the Closest Pair Problem". In: *Journal of the ACM* 62.2 (2015), p. 13.

[Vil+08]   Cédric Villani et al. *Optimal transport: old and new*. Vol. 338. Springer, 2008.

[Wil18]   Ryan Williams. "On the difference between closest, furthest, and orthogonal pairs: Nearly-linear vs barely-subquadratic complexity". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2018, pp. 1207–1215.

[Zuz23]   Goran Zuzic. "A Simple Boosting Framework for Transshipment". In: *31st Annual European Symposium on Algorithms (ESA 2023)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. 2023, pp. 104–1.