Efficient Approximation Algorithm for Computing Wasserstein Barycenter under Euclidean Metric

Pankaj K. Agarwal* Sharath Raghvendra† Pouyan Shirzadian‡ Keegan Yao*

Abstract

Given a set of probability distributions, the *Wasserstein barycenter* problem asks to compute a distribution that minimizes the average Wasserstein distance, or optimal transport cost, from all the input distributions. Wasserstein barycenters preserve common geometric features of the input distributions, making them useful in machine learning and data analytics tasks.

We present a near-linear time algorithm that computes the Wasserstein barycenter within a relative $(1+\varepsilon)$-approximation in fixed dimensions. We are not aware of an efficient relative-approximation algorithm for this problem even in two dimensions. To obtain our results, we first present a near-linear-time dynamic-programming-based primal-dual algorithm to compute an optimal Wasserstein barycenter under a tree metric. We then combine our tree-based algorithms with the boosting framework to obtain a $(1+\varepsilon)$-approximation in near-linear time.

1 Introduction

Optimal transport (OT) is a popular method for comparing a pair of distributions based on the displacement of mass, and thus the OT cost depends on the shape of the distributions. OT has been widely studied across many disciplines, including mathematics, statistics, economics, engineering, machine learning, and computer vision; see the books [35, 40] and recent survey papers [24, 33]. However in many of these applications, the number of distributions is large and one wants to compute a representative distribution of these distributions, or cluster them and compute a representative distribution for each cluster. This raises the problem of computing an average distribution among a group of distributions, which is referred to as the *Wasserstein barycenter*. The barycenter problem is then the natural extension of displacement interpolation to more than two distributions.

Despite significant progress on computing optimal transport efficiently, the barycenter problem remains computationally challenging and relatively little is known. This paper aims to make progress in this direction.

1.1 Problem statement. Let $\mu\colon A \to [0,1]$ and $\nu\colon B \to [0,1]$ be two discrete probability distributions with finite support $A, B \subset \mathbb{R}^d$, respectively. Additionally let $||x-y||$ be the Euclidean distance between pairs of points x, y in $\mathbb{R}^d$. A *discrete transport plan* is a mapping $f\colon A \times B \to \mathbb{R}_{\geq 0}$ that assigns the mass transported along each edge $(a,b) \in A \times B$ such that $\sum_{b\in B} f(a,b) = \mu(a)$ for each point $a \in A$, and $\sum_{a\in A} f(a,b) = \nu(b)$ for each point $b \in B$. The cost of f is given by $¢(f) = \sum_{(a,b)\in A\times B} f(a,b)||a-b||$. The *discrete optimal transport (OT) problem* asks for a discrete transport plan f with the minimum cost. We refer to such a minimum cost plan as an *OT plan*, and to the cost of the OT plan, denoted by $w(\mu,\nu)$, as the 1-*Wasserstein distance* (or *earth mover's distance*) between μ and ν, i.e. $w(\mu,\nu) = \min_f ¢(f)$. In general, for a fixed $p \geq 1$, one can define $¢(f) = \left(\sum_{(a,b)\in A\times B} f(a,b)||a-b||^p\right)^{\frac{1}{p}}$ and the cost of the cheapest plan under this cost function is referred to as the p-Wasserstein distance between μ and ν. In this paper, we focus on $p = 1$. There is much work for the case of $p = 2$ as well; see [40].

Given a family $\boldsymbol{\mu} = \langle \mu_1, \ldots, \mu_k \rangle$ of k discrete probability distributions $\mu_i\colon X_i \to [0,1]$ for $i = 1, \ldots, k$, where $X_i \subseteq \mathbb{R}^d$ for each $i \leq k$ and $\left|\bigcup_{i=1}^k X_i\right| \leq n$, the *Wasserstein barycenter* (or simply *barycenter* for brevity) of $\boldsymbol{\mu}$ is a probability distribution $\mu^*\colon X \to [0,1]$, where $X \subset \mathbb{R}^d$ is a finite point set, such that $\sum_{i=1}^k w(\mu^*, \mu_i)$ is

*Department of Computer Science, Duke University.
†Department of Computer Science, North Carolina State University.
‡Department of Computer Science, Virginia Tech.

minimized[1]. The *barycenter problem* asks for computing μ^* along with optimal transport plans $\boldsymbol{f}^* = \langle f_1^*, \ldots, f_k^* \rangle$, where f_i^* is a transport plan from μ_i to μ^*. Define $\Gamma(\mu^*) = ¢(\boldsymbol{f}^*) := \sum_{i=1}^k ¢(f_i^*) = \sum_{i=1}^k w(\mu^*, \mu_i)$. A probability distribution $\widetilde{\mu}$ is called a $(1+\varepsilon)$-*approximate barycenter* if $\Gamma(\widetilde{\mu}) \leq (1+\varepsilon) \cdot \Gamma(\mu^*)$.

Reduction to flow-like problem. Let $G = (V, E)$ with $|V| = n$ be an undirected weighted graph, where $c\colon E \to \mathbb{R}_{\geq 0}$ denotes the cost function. Let $\mu_i\colon V \to [0,1]$, for $1 \leq i \leq k$, be k probability distributions on G. The barycenter problem over the graph G asks for a probability distribution $\mu\colon V \to [0,1]$ where the average optimal transport cost from μ to each of the distributions μ_i is minimized. To reduce the barycenter problem to the flow problem, we replace each edge $(u,v) \in E$ with directed edges $u \to v$ and $v \to u$ and define flow functions $\boldsymbol{f} = \langle f_1, \ldots, f_k \rangle$ on this directed graph. Then, computing an optimal barycenter distribution μ along with the flow functions $\boldsymbol{f}$ can be formulated as the following LP:

$$\begin{aligned} \min_{\mu, \boldsymbol{f}=\langle f_1,\ldots,f_k\rangle} & \sum_i \sum_{(u\to v)\in E} f_i(u \to v) c(u,v) \\ \text{s.t.} \quad & \sum_{u\to v\in E} f_i(u \to v) - \sum_{v\to u\in E} f_i(v \to u) = \mu_i(u) - \mu(u) \ \ \forall u \in V, i \leq k, \\ & \sum_u \mu(u) = 1, \ \ f_i, \mu \geq 0. \end{aligned}$$

Recall that $¢(f_i) = \sum_{(u,v)\in E} f_i(u \to v) c(u,v)$ denotes the cost of f_i. Then the primal objective function is $¢(\boldsymbol{f}) = \sum_i ¢(f_i)$. The dual of the above LP is

$$\begin{aligned} \max_{\boldsymbol{\phi}, \lambda} & \sum_i \sum_{u\in V} \phi_i(u)\mu_i(u) + \lambda \\ \text{s.t.} \ \ & \phi_i(u) - \phi_i(v) \leq c(u,v) \ \ \forall (u,v) \in E, \\ & \sum_i \phi_i(v) + \lambda \leq 0 \ \ \forall v \in V. \end{aligned}$$

Again, we use $\boldsymbol{\phi} = \langle \phi_1, \ldots, \phi_k \rangle$ to denote the family of dual functions ϕ_i for each index i. The key difference between the dual barycenter problem and the dual LP for min-cost flow in a graph lies in the constraints $\sum_i \phi_i(v) + \lambda \leq 0$.

1.2 Related work. The discrete OT problem is a special case of the minimum-cost flow problem on complete bipartite graphs, and therefore can be solved exactly using primal-dual algorithms in $O(n^3 \log n)$ time [34]. With the recent breakthroughs on the minimum-cost flow problem, the OT problem can also be solved in $n^{2+o(1)}\mathrm{poly}(\log(\Delta), \log(U))$ time, where Δ is the spread, i.e., the ratio of the distance of the farthest to the closest pair of points in $A \cup B$, and U depends on the minimum non-zero values of μ and ν [16]. An ε-OT under the Euclidean metric in fixed dimension can be computed in $O(n\varepsilon^{-2d-5} \log n \log^{2d+5} \log n)$ time with constant probability [2] or in $O(n\varepsilon^{-d-2} \log^5 n \log\log n)$ time deterministically [21]. For higher dimensions, one can use similar techniques as in [2, 21] to compute $2c$-approximate OT plans in $d^3 n^{1+\frac{1}{c^2}} \log^{O(1)} n$ time. Recently a subquadratic algorithm for computing 2-OT was proposed in [7]. There is also much work on computing an additive approximation of OT [3, 18, 27, 32].

In contrast to the OT problem, which is known to be solvable exactly in polynomial time, the barycenter problem is known to be NP-hard in arbitrary dimensions [5] and can be computed within additive error $\varepsilon > 0$ in $(nk)^{O(d)}\mathrm{poly}\log \varepsilon^{-1}$ time if d is constant [4]. We are unaware of any efficient (relative-)approximation algorithm for the barycenter problem, but an ε-additive approximation can be computed in $O(n^2 k \varepsilon^{-2})$ time [11, 19, 26]. As for the OT problem, much of the work on additive approximations study regularized versions of the barycenter problem. There is also a slew of results on other first-order numerical algorithms such as accelerated gradient

[1]Strictly speaking, we should take the infimum over an arbitrary distribution possibly with continuous support. It is known, however, that if each input support is a set of n points, then even if the Wasserstein barycenter objective is taken with respect to a continuous distribution, there exists an optimal barycenter distribution with a finite support of size $O(n^k)$ [6].

descent and stochastic gradient descent, which in some cases provide comparable convergence guarantees for ε-additive approximations [17, 20, 22, 30]. Finally, we refer the reader to [13, 28, 36, 39] for several fast heuristics. We emphasize that much of the existing work studies the barycenter problem when the support of the barycenter is considered fixed [31].

Computing the Wasserstein distance under tree metrics can be expressed as computing the ℓ_1-distance between two points [15, 23, 29]. This suggests that computing the barycenter under tree metrics could be reduced to computing the ℓ_1 median of a set of points. This approach was studied experimentally [28]. However, this is not necessarily true, i.e., if we use the ℓ_1-embedding of input distributions, the barycenter is not necessarily the ℓ_1-median of the resulting points.

1.3 Summary of results. Our main result (Section 3) is the first near-linear time algorithm for computing a $(1+\varepsilon)$-approximate barycenter in any fixed dimensional Euclidean space:

THEOREM 1.1. *Let $\boldsymbol{\mu} = \langle \mu_1, \ldots, \mu_k \rangle$ be a family of k discrete probability distributions with finite support sets $X_i \subset \mathbb{R}^d$ for some fixed $d \geq 1$, and $\varepsilon > 0$ a parameter. Set $|\bigcup_i X_i| = n$. Then a discrete probability distribution $\widetilde{\mu}$ of support size $O(n\varepsilon^{-d} \log n)$ can be computed in $O\left(nk\varepsilon^{-2d-4} \log^5 n \log(nk) \log\log n\right)$ time that is a $(1+\varepsilon)$-approximate barycenter with probability at least $1/2$.*

A key step of our algorithm is a near-linear exact primal-dual algorithm for computing the barycenter of a family of distributions under tree metrics (Section 2), which is of independent interest.

THEOREM 1.2. *Let $T = (V, E)$ be a tree with $|V| = n$, height h, and edge costs $c\colon E \to \mathbb{R}_{\geq 0}$. Given a collection of k distributions $\mu_i\colon V \to [0,1]$ for $1 \leq i \leq k$, an optimal barycenter $\mu^*\colon V \to [0,1]$ along with corresponding flow functions and dual functions can be computed in $O(nk \min\{\log^2 n, h\})$ time.*

While OT under tree metrics can be computed by a simple greedy algorithm, this approach does not extend to the barycenter problem. We therefore propose a dynamic programming based primal-dual algorithm that not only computes μ^* and the transport maps between μ^* and μ_i's, but also computes an optimal dual solution. Our approach can be thought of as constructing all k transport plans simultaneously by modifying a feasible solution via alternating paths that reduce the barycenter cost. The tree structure then forces the alternating paths for each distribution to be identical if one wishes to preserve feasibility. This is not true in arbitrary graphs, where it is possible to get stuck at a local minimum if one forces alternating paths for all distributions to be identical. With the proper choice of initial flows and a feasible probability distribution, one can use this structure to construct good alternating paths quickly and guarantee only a small number of paths are needed. Once an optimal primal solution is computed, an optimal dual solution can be computed by a single additive update rule along edges through a tree traversal starting from one source point of the optimal barycenter distribution μ^*.

Given Theorem 1.2, we compute a $(1+\varepsilon)$-approximate barycenter under the Euclidean metric in three stages. First, adapting the ideas from the construction of approximate Voronoi diagrams [8, 10], we construct a set $V \supseteq X$ of $O(n\varepsilon^{-d})$ points that is a good candidate set for the support of a $(1+\varepsilon)$-approximate barycenter. In particular, our candidate support size is near-linear in n for fixed dimension d and does not depend on the diameter of the point set. Next, we construct a hierarchical graph $\mathcal{G} = (V, E)$ that is a $(1+\varepsilon)$-spanner of V in expectation and that contains a randomized tree $\mathcal{T} = (V, E')$ (a variant of a randomly shifted quadtree) as its subgraph such that $\mathcal{T}$ is an $O(\log n)$-spanner of $\mathcal{G}$. Finally, we use a multiplicative weight update (MWU) method for computing a $(1+\varepsilon)$-approximate barycenter for the shortest path metric on $\mathcal{G}$. Our approach is an extension of the MWU algorithms developed for minimum-cost flows [37, 41]. We use our tree algorithm at each step of the MWU algorithm as the oracle.

2 Computing Barycenters in Trees

Let $T = (V, E)$ be a tree with $|V| = n$, let $c\colon E \to \mathbb{R}_{\geq 0}$ be a cost function, and let $\mu_i\colon V \to [0,1]$, for $1 \leq i \leq k$, be k probability distributions on V. For simplicity, we assume the tree T to be a binary tree. Otherwise, any tree can be converted to a binary tree, potentially at the cost of increasing its height. Let h be the height of T. For a non-root node $u \in T$, let $\mathsf{p}(u)$ denote its parent in T, and let $\mathsf{C}[u]$ denote the children of u. For a pair of nodes $u, v \in T$, let $u \rightsquigarrow v$ denote the unique path from u to v in T, and let $\mathrm{lca}(u, v)$ denote their least common ancestor in T.

We describe a primal-dual barycenter algorithm TREE-BARYCENTER that computes, in near-linear time, a barycenter distribution μ^* of $\boldsymbol{\mu} = \langle \mu_1, \ldots, \mu_k \rangle$ on T, corresponding transport plans $\boldsymbol{f} = \langle f_1, \ldots, f_k \rangle$ from μ^* to μ_i, and dual weights corresponding to each input distribution. We first prove a few properties of the barycenter and the corresponding optimal flows on T. Next, we describe a dynamic programming based algorithm to compute the optimal solution using these properties that runs in $O(nkh)$ time, where h is the height of T. By using the link-cut tree [38], we improve the runtime to $O(nk \min\{h, \log^2 n\})$.

2.1 Properties of an optimal solution.

We begin by proving a few properties of the optimal flow in T.

Circulation. Given a feasible solution $(\mu, \boldsymbol{f})$ for T, define a *circulation* to be a collection of functions $g_i \colon E \to \mathbb{R}$ for $1 \le i \le k$ such that $f_i + g_i$ for $1 \le i \le k$ is a feasible solution to the barycenter problem. By standard use of flow conservation constraints, it can be shown that a feasible collection of flows f_i, $1 \le i \le k$, and distribution μ is a barycenter if and only if there do not exist circulations $g_i \colon E \to \mathbb{R}$, for $1 \le i \le k$, such that $\sum_i \cent(f_i + g_i) < \cent(\boldsymbol{f})$. In fact, this property holds for the barycenter problem over any graph. One obtains the following stronger guarantee for trees, which is the key property, using the fact that trees have single-edge cuts.

LEMMA 2.1. *A feasible collection of flows f_i, $1 \le i \le k$, and distribution μ is a tree barycenter if and only if there does not exist a circulation $g \colon E \to \mathbb{R}$ such that the flows $f_i' := f_i + g$ for $1 \le i \le k$ are feasible and $\sum_i \cent(f_i + g) < \cent(\boldsymbol{f})$.*

Proof. Suppose $\boldsymbol{f} = \langle f_1, \ldots, f_k \rangle$ and $\boldsymbol{f'} = \langle f_1', \ldots, f_k' \rangle$ are two distinct feasible solutions to the flow problem. Define the circulation $g_i = f_i' - f_i$. To prove this lemma, it suffices to show that if the network $G = (V, E)$ is a tree, then $g_i = g_j$ for all i, j. Since f_i is a feasible solution, for all i, j and all $u \in V$,

$$\sum_{v \to u \in E} f_i(v \to u) - \sum_{u \to v \in E} f_i(u \to v) + \mu_i(u) = \sum_{v \to u \in E} f_j(v \to u) - \sum_{u \to v \in E} f_j(u \to v) + \mu_j(u).$$

That is, there is a probability distribution μ induced by the flow f_i and the constraints of the linear program. The analogous statement for f_i', f_j' holds for all i, j and all $u \in V$. Let $a \to b$ be an arbitrary edge in the tree G, and let (A, B) be the cut defined by the edge $a \to b$ where $a \in A$ and $b \in B$. Summing over vertices in A, for any $1 \le i \le k$,

$$\begin{aligned}
&\sum_{u \in A} \left(\sum_{v \to u \in E} f_i(v \to u) - \sum_{u \to v \in E} f_i(u \to v) + \mu_i(u) \right) \\
&\quad = \sum_{u \in A} \left(\sum_{v \in B : v \to u \in E} f_i(v \to u) - \sum_{v \in B : u \to v \in E} f_i(u \to v) + \mu_i(u) \right) \\
&\quad = f_i(b \to a) - f_i(a \to b) + \sum_{u \in A} \mu_i(u),
\end{aligned}$$

where the first equality holds due to the cancellation of edges $u \to v$ in the summation where both u and v are in A. Hence, for all $1 \le i, j \le k$,

$$f_i(b \to a) - f_i(a \to b) + \sum_{u \in A} \mu_i(u) = f_j(b \to a) - f_j(a \to b) + \sum_{u \in A} \mu_j(u). \tag{2.1}$$

The analogous computation for f_i' gives

$$f_i'(b \to a) - f_i'(a \to b) + \sum_{u \in A} \mu_i(u) = f_j'(b \to a) - f_j'(a \to b) + \sum_{u \in A} \mu_j(u) \tag{2.2}$$

for all i, j since $\boldsymbol{f'}$ is also a feasible solution. By taking the difference of (2.1) and (2.2),

$$\begin{aligned}
&\left(f_i'(b \to a) - f_i'(a \to b) + \sum_{u \in A} \mu_i(u) \right) - \left(f_i(b \to a) - f_i(a \to b) + \sum_{u \in A} \mu_i(u) \right) \\
&\quad = \left(f_j'(b \to a) - f_j'(a \to b) + \sum_{u \in A} \mu_j(u) \right) - \left(f_j(b \to a) - f_j(a \to b) + \sum_{u \in A} \mu_j(u) \right).
\end{aligned}$$

Canceling $\sum_{u\in A}\mu_i(u)$ and substituting $g_i(b\to a) = f_i'(b\to a) - f_i(b\to a)$ gives the expression

$$g_i(b\to a) - g_i(a\to b) = g_j(b\to a) - g_j(a\to b)$$

for all i, j. Note the edge $a\to b$ was chosen arbitrarily, so this holds for all edges $a\to b\in E$. ∎

Net cost of a path. The above lemma suggests that if f_i's are not optimal flows, then we can improve the solution by transporting the same mass of all distributions across each edge. We next define the net cost of transporting additional mass on the edge $u\to v$, which captures the rate of change in the cost of flows f_i when we transport an additional infinitesimal mass of all distributions along the directed edge $u\to v$ (if the mass transported along the edge $v\to u$ is positive, then we decrease the flow).

Given a feasible family of flows $\boldsymbol{f} = \langle f_1,\dots,f_k\rangle$, define two sets $K_1^{\boldsymbol{f}}(u\to v)$ and $K_2^{\boldsymbol{f}}(u\to v)$ for each directed edge $u\to v$ of T as $K_1^{\boldsymbol{f}}(u\to v) := \{i : f_i(u\to v) > 0\}$ and $K_2^{\boldsymbol{f}}(u\to v) := \{i : f_i(u\to v) = 0\}$. These sets describe the distribution indices where flow along edge $u\to v$ is positive and zero, respectively. Additionally define $k_j^{\boldsymbol{f}}(u\to v) := |K_j^{\boldsymbol{f}}(u\to v)|$ for $j = 1, 2$. Note that $k_2^{\boldsymbol{f}}(u\to v) = k_2^{\boldsymbol{f}}(v\to u)$ and $k_1^{\boldsymbol{f}}(v\to u) = k - k_1^{\boldsymbol{f}}(u\to v) - k_2^{\boldsymbol{f}}(u\to v)$. The change in cost after transporting mass along $u\to v$ can be broken into two components:

1. $\alpha^{\boldsymbol{f}}(u\to v) := \left(k_1^{\boldsymbol{f}}(u\to v) - k_1^{\boldsymbol{f}}(v\to u)\right)\cdot c(u,v)$, which is the change in the cost by adjusting the flow to the distributions that already committed to a direction along the edge $u\to v$, and
2. $\beta^{\boldsymbol{f}}(u\to v) := k_2^{\boldsymbol{f}}(u\to v)\cdot c(u,v)$, which is the change in cost of adjusting flow to the distributions that have no flow along $u\to v$.

Using $\alpha^{\boldsymbol{f}}$ and $\beta^{\boldsymbol{f}}$, we define the *net cost* $\theta^{\boldsymbol{f}}(u\to v)$ of the edge $u\to v$ as

$$\theta^{\boldsymbol{f}}(u\to v) := \alpha^{\boldsymbol{f}}(u\to v) + \beta^{\boldsymbol{f}}(u\to v) = (k - 2k_1^{\boldsymbol{f}}(v\to u))\cdot c(u,v).$$

If $\boldsymbol{f} = \langle f_1,\dots,f_k\rangle$ is clear from context, we drop the superscript $\boldsymbol{f}$ for brevity. Note that θ is not skew symmetric; that is, $\theta(u\to v)\neq -\theta(v\to u)$. However,

$$\theta(v\to u) = -\alpha(u\to v) + \beta(u\to v). \tag{2.3}$$

A key observation is that θ is monotone in the following sense.

LEMMA 2.2. *Suppose $\boldsymbol{f}' = \langle f_1',\dots,f_k'\rangle$ and $\boldsymbol{f} = \langle f_1,\dots,f_k\rangle$ are two feasible flows for the barycenter problem on tree $T = (V, E)$. For any $u\to v\in E$, if $f_i'(u\to v)\geq f_i(u\to v)$ for all $i\leq k$, then $\theta^{\boldsymbol{f}'}(u\to v)\geq\theta^{\boldsymbol{f}}(u\to v)$.*

Proof. If $f_i'(u\to v)\geq f_i(u\to v)$ for all i, then we observe that $K_1^{\boldsymbol{f}'}(v\to u)\subseteq K_1^{\boldsymbol{f}}(v\to u)$ by the definition of K_1. Therefore, $k_1^{\boldsymbol{f}'}(v\to u)\leq k_1^{\boldsymbol{f}}(v\to u)$ and $k_1^{\boldsymbol{f}'}(u\to v) + k_2^{\boldsymbol{f}'}(u\to v)\geq k_1^{\boldsymbol{f}}(u\to v) + k_2^{\boldsymbol{f}}(u\to v)$. The result follows from the definition of net cost. ∎

Negative cost augmenting path. Let $P = u\rightsquigarrow v$ be the path from u to v in T. We define the *net cost* of P to be $\theta(P) := \sum_{x\to y\in P}\theta(x\to y)$. P is called a *negative net-cost* path if $\theta(P) < 0$. Furthermore, given a feasible solution $(\mu, \boldsymbol{f})$ to the barycenter problem, define an *augmenting path* to be a path $P = \langle u_1,\dots,u_t\rangle$ where $\mu(u_1) > 0$. Building on Lemma 2.1, we prove the following property of an optimal solution.

LEMMA 2.3. *A feasible solution $(\mu, \boldsymbol{f})$ is a barycenter of $\langle\mu_1,\dots,\mu_k\rangle$ if and only if there is no negative net-cost augmenting path in T.*

Proof. If there exists a negative net cost augmenting path $u\rightsquigarrow w$ in T, then by the definition of an augmenting path, $\mu(u) > 0$. Define

$$\delta := \min\left\{\mu(u), \min_{x\to y\in u\rightsquigarrow w}\ \min_{i\in K_1(y\to x)} f_i(y\to x)\right\}$$

as the amount of additional mass that can be transported through the path $u \rightsquigarrow w$ before the net cost $\theta(u \rightsquigarrow w)$ changes. Note that $\delta > 0$, by the definitions of K_1 and augmenting paths. Define

$$g(x \to y) = \begin{cases} \delta, & x \to y \in u \rightsquigarrow w, \\ -\delta, & y \to x \in u \rightsquigarrow w, \\ 0, & \text{otherwise.} \end{cases}$$

Then

$$\sum_i \text{¢}(f_i + g) - \text{¢}(\boldsymbol{f}) = \sum_{x \to y \in u \rightsquigarrow w} \theta(x \to y) \cdot \delta = \delta \cdot \theta(u \rightsquigarrow w) < 0,$$

which means $\sum_i \text{¢}(f_i + g) < \text{¢}(\boldsymbol{f})$ and by Lemma 2.1, $(\mu, \boldsymbol{f})$ is not an optimal solution. Therefore, if $(\mu, \boldsymbol{f})$ is a barycenter of $\langle \mu_1, \ldots, \mu_k \rangle$, then there are no negative net cost augmenting paths in T.

Next, suppose $(\mu, \boldsymbol{f})$ is a feasible solution that has no negative net cost augmenting paths in T. Let $(\mu^*, \boldsymbol{f^*})$ be an optimal solution where $\mu^* \neq \mu$. Let $g = f_i^* - f_i$ denote the circulation deforming $\boldsymbol{f}$ into $\boldsymbol{f^*}$. We note that by Lemma 2.1, $f_i^* - f_i = f_j^* - f_j$ for all i, j, i.e. g is a single flow function. By standard flow decomposition, g can be decomposed into a sequence of flows $g^1, \ldots, g^z$ for some $z \geq 1$, where each g^i routes some $\delta_i > 0$ mass along a path $u_i \rightsquigarrow v_i$, $g = \sum_i g^i$, and for every edge $u \to v$, if $g^i(u \to v) > 0$ for some i, then $g^j(u \to v) \geq 0$ for all j (i.e. all paths in the path decomposition route flow in the same direction along every edge). By the assumption that there are no negative net cost augmenting paths in T, we get $\theta(u_i \rightsquigarrow v_i) \geq 0$ for all i. Since θ is non-decreasing by Lemma 2.2 and the construction of the path decomposition (direction along every edge is the same), after augmenting $\boldsymbol{f}$ by the circulation g,

$$\text{¢}(\boldsymbol{f^*}) \geq \text{¢}(\boldsymbol{f}) + \sum_{i=1}^{z} \theta(u_i \rightsquigarrow v_i) \cdot \delta_i \geq \text{¢}(\boldsymbol{f}),$$

where the first inequality follows from the fact that θ is non-decreasing after augmenting along each path (Lemma 2.2) and the second inequality follows from the assumption that $\theta(u_i \rightsquigarrow v_i) \geq 0$ for all i. Recall that $(\mu^*, \boldsymbol{f}^*)$ is an optimal feasible solution; therefore, $(\mu, \boldsymbol{f})$ is an optimal feasible solution ∎

Lemma 2.3 immediately gives an algorithm for computing a barycenter, analogous to a min-cost flow algorithm. We start with an arbitrary feasible solution, repeatedly find negative net-cost augmenting paths, transport the mass of all distributions along that path, and repeat. This algorithm may not converge quickly, so we prove an additional property.

Minimum net-cost augmenting paths. Suppose we initially compute a feasible flow $\boldsymbol{f}$ by pushing all mass up the tree to its root r. For every edge $u \to v$ in T where u is a child of v, set the flow $f_i(u \to v) = \mu_i(u)$ if u is a leaf and $f_i(u \to v) = \mu_i(u) + \sum_{w \in \mathsf{C}[u]} f_i(w \to u)$ otherwise; here, $\mathsf{C}[u]$ denotes the set of children of u. Set $\mu(r) = 1$. While there exists a negative net cost augmenting path, we send mass from every distribution through the minimum net-cost augmenting path $u \rightsquigarrow v$ until either (i) $\mu(u)$ becomes 0, or (ii) $k_1(y \to x)$ decreases for some edge $x \to y$ on $u \rightsquigarrow v$. We prove that the minimum net-cost augmenting path always starts from the root of T to a node, i.e., we always augment mass down the tree from its root.

Let $P_1, P_2, \ldots$ be the sequence of minimum (negative) net-cost augmenting paths computed by the algorithm. We prove each P_i is a downward path by induction on the number of steps. Since initially, only the root has non-zero mass, P_1 is obviously a downward path starting at the root. Suppose each P_t, for all $t < t'$, is a downward path from the root. Then we prove that $P_{t'}$ is also a downward path from the root.

Suppose to contrary, $P_{t'} = u \rightsquigarrow v$ is not a downward path starting from the root, i.e., $u \neq r$. Let $w = \text{lca}(u, v)$ be the least common ancestor of u and v. Note that w can be u or v itself. Since u has non-zero mass and initially $\mu(u) = 0$, there was an augmenting path P_{t_0} for $1 \leq t_0 < t'$ that pushed the mass from r to u. Let $t_0 < t'$ be the last time such that $P_{t_0} = r \rightsquigarrow u$. We use the following additional time-dependent notations. Since $K_1(\cdot), K_2(\cdot), \alpha(\cdot), \beta(\cdot)$, and $\theta(\cdot)$ vary over time, we use $K_1^t(\cdot), K_2^t(\cdot), \alpha^t(\cdot), \beta^t(\cdot)$, and $\theta^t(\cdot)$ to denote their values in the beginning of step t. We will be interested in time steps $t < t'$ when the augmenting path P_t passed through an edge of the path $w \rightsquigarrow u$. For an edge $x \to y$ with y being the child of x, let $\tau(x \to y)$ denote the max value of $t < t'$ for which P_t contained $x \to y$, i.e.,

$$\tau(x \to y) = \max\{t < t' : x \to y \in P_t\}.$$

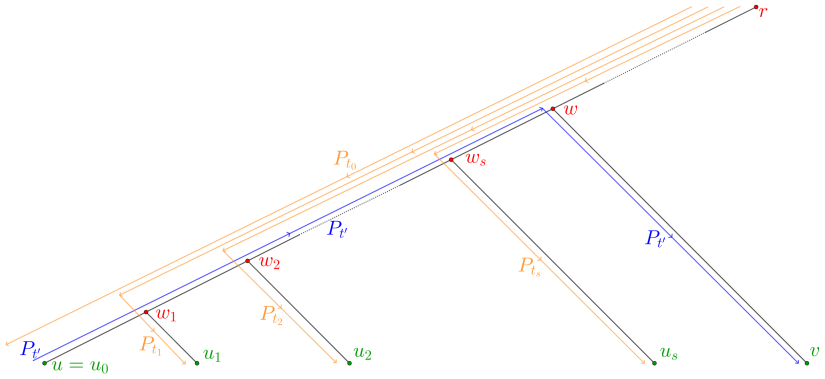

Figure 1: An example of the paths P_{t_i} with corresponding endpoints u_i and branching points w_i, for $0 \le i \le s$.

We also define $\tau(y \to x) = \tau(x \to y)$. For simplicity, we define

$$\psi(x \to y) = \theta^{\tau(x \to y)}(x \to y),$$

i.e., its net cost just before the last time a flow was augmented through it prior to time t'. For a path P in T, define $\psi(P) = \sum_{x \to y \in P} \psi(x \to y)$. Let $t_0 < t_1 < t_2 < \cdots < t_s < t'$ be the time instances such that $t_i = \tau(x \to y)$ for some edge $x \to y \in w \rightsquigarrow u$; $t_0 \le t_i \le t'$. Suppose $P_{t_i} = r \rightsquigarrow u_i$ for $1 \le i \le s$, and let $w_i = \mathrm{lca}(u, u_i)$. Set $u = u_0, w_0 = u$, and $w_{s+1} = w$. Then w_i is a descendant of w_{i+1}, and all of them are descendants of $w = w_{s+1}$; see Figure 1. Therefore $r \rightsquigarrow w_i \subseteq P_{t_0} \cap P_{t_i}$ and $w_i \rightsquigarrow u_i = P_{t_i} \setminus P_{t_0}$. Furthermore, by definition of $\psi(\cdot)$,

$$\psi(w_{\ell+1} \rightsquigarrow w_\ell) = \theta^{t_\ell}(w_{\ell+1} \rightsquigarrow w_\ell), \quad 0 \le \ell \le s. \tag{2.4}$$

We now prove a sequence of lemmas that will contradict the assumption that P_{t_s} is the minimum net-cost path at t_s.

LEMMA 2.4. *Let u, v be two nodes in T with u being an ancestor of v. Then for any $t < t'$,*

(i) $\theta^t(u \rightsquigarrow v) \le \theta^{t+1}(u \rightsquigarrow v)$,

(ii) $\theta^t(v \rightsquigarrow u) \le \theta^{t+1}(v \rightsquigarrow u)$, *and*

(iii) if the augmenting path P_t passed through v, then $\theta^t(u \rightsquigarrow v) = -\theta^{t+1}(v \rightsquigarrow u)$.

Proof. Parts (i) and (ii) follow immediately from Lemma 2.2 because all augmenting paths before t' are downward. To prove part (iii), we observe that at time t, mass was transported along the path $u \rightsquigarrow v$. Therefore, any flow which was zero along an edge of $u \rightsquigarrow v$ becomes positive in this direction, and any flow which became zero along an edge of $u \rightsquigarrow v$ must have been canceled with some positive flow in the $v \rightsquigarrow u$ direction. We conclude that for every edge $x \rightsquigarrow y$ on the path $u \rightsquigarrow v$,

1. $K_1^{t+1}(x \to y) = K_1^t(x \to y) \cup K_2^t(x \to y)$, and
2. $K_1^{t+1}(y \to x) \cup K_2^{t+1}(x \to y) = K^t(y \to x)$.

By definition of θ, α, β, and the above properties, we have

$$\begin{aligned}
\theta^t(x \to y) &= \alpha^t(x \to y) + \beta^t(x \to y) \\
&= ((k_1^t(x \to y) - k_1^t(y \to x)) + (k_2^t(x \to y))) \cdot c(x \to y) \\
&= ((k_1^t(x \to y) + k_2^t(x \to y)) - (k_1^t(y \to x))) \cdot c(x \to y) \\
&= ((k_1^{t+1}(x \to y)) - (k_1^{t+1}(y \to x) + k_2^{t+1}(y \to x))) \cdot c(x \to y) \\
&= -\alpha^{t+1}(y \to x) - \beta^{t+1}(y \to x) \\
&= -\theta^{t+1}(y \to x)
\end{aligned}$$

for every edge $x \to y$ on the path $u \rightsquigarrow v$. The lemma now follows. ∎

LEMMA 2.5. $\psi(w \rightsquigarrow u) \leq \theta^{t_s}(w \rightsquigarrow u_s)$.

Proof. We claim that for any $0 \leq \ell \leq s$,

$$\psi(w \rightsquigarrow w_\ell) + \theta^{t_\ell}(w_\ell \rightsquigarrow u_\ell) \leq \psi(w \rightsquigarrow w_{\ell+1}) + \theta^{t_{\ell+1}}(w_{\ell+1} \rightsquigarrow u_{\ell+1}).$$

Assuming this claim is true, then by induction on ℓ and using (2.4), we obtain

$$\begin{aligned}
\psi(w \rightsquigarrow u) &\leq \psi(w \rightsquigarrow w_s) + \theta^{t_s}(w_s \rightsquigarrow u_s) \\
&= \theta^{t_s}(w \rightsquigarrow w_s) + \theta^{t_s}(w_s \rightsquigarrow u_s) \quad \text{(by (2.4))} \\
&= \theta^{t_s}(w \rightsquigarrow u_s).
\end{aligned}$$

To prove the claim, we observe that $\theta^{t_\ell}(w_{\ell+1} \rightsquigarrow u_\ell) \leq \theta^{t_\ell}(w_{\ell+1} \rightsquigarrow u_{\ell+1})$ because we chose $r \rightsquigarrow u_\ell$ instead of $r \rightsquigarrow u_{\ell+1}$ in step t_ℓ. Then it follows that

$$\begin{aligned}
\psi(w \rightsquigarrow w_\ell) + \theta^{t_\ell}(w_\ell \rightsquigarrow u_\ell) &\leq \psi(w \rightsquigarrow w_{\ell+1}) + \psi(w_{\ell+1} \rightsquigarrow w_\ell) + \theta^{t_\ell}(w_\ell \rightsquigarrow u_\ell) \\
&= \psi(w \rightsquigarrow w_{\ell+1}) + \theta^{t_\ell}(w_{\ell+1} \rightsquigarrow u_\ell) \quad \text{(by (2.4))} \\
&\leq \psi(w \rightsquigarrow w_{\ell+1}) + \theta^{t_\ell}(w_{\ell+1} \rightsquigarrow u_{\ell+1}) \\
&\leq \psi(w \rightsquigarrow w_{\ell+1}) + \theta^{t_{\ell+1}}(w_{\ell+1} \rightsquigarrow u_\ell) \quad \text{(by Lemma 2.4(i))}.
\end{aligned}$$

This completes the proof. ∎

Using the above two lemmas, we obtain the following.

LEMMA 2.6. *If $u \rightsquigarrow v$ is a negative net-cost path at time t', where u is not a root and the minimum net-cost path P_t starts at the root for all $t < t'$, then $\theta^{t_s}(r \rightsquigarrow v) < \theta^{t_s}(r \rightsquigarrow u_s)$.*

Proof. Since $u \rightsquigarrow v$ is a negative net-cost path at time t', $\theta^{t'}(u \rightsquigarrow v) < 0$. Since no augmenting path P_t for $t > t_\ell, 0 \geq \ell \geq s$, touches $w_{\ell+1} \rightsquigarrow w_\ell$ and P_{t_ℓ} contains $w_{\ell+1} \rightsquigarrow w_\ell$, by Lemma 2.5, $\theta^{t'}(u \rightsquigarrow w) = -\psi(w \rightsquigarrow u)$. Therefore

$$\begin{aligned}
\theta^{t'}(u \rightsquigarrow v) &= \theta^{t'}(u \rightsquigarrow w) + \theta^{t'}(w \rightsquigarrow v) \\
&\geq -\psi(w \rightsquigarrow u) + \theta^{t'}(w \rightsquigarrow v) \quad \text{(by Lemma 2.4)} \\
&\geq -\theta^{t_s}(w \rightsquigarrow u_s) + \theta^{t'}(w \rightsquigarrow v) \quad \text{(by Lemma 2.5)} \\
&= -\theta^{t_s}(r \rightsquigarrow w) - \theta^{t_s}(w \rightsquigarrow u_s) + \theta^{t_s}(r \rightsquigarrow w) + \theta^{t'}(w \rightsquigarrow v) \\
&= -\theta^{t_s}(r \rightsquigarrow u_s) + \theta^{t_s}(r \rightsquigarrow v).
\end{aligned}$$

The lemma now follows because $\theta^{t'}(u \rightsquigarrow v) < 0$. ∎

The above lemma implies that if u is not the root of T then the minimum net-cost path at time t_s was not $r \rightsquigarrow u_s$ but instead $r \rightsquigarrow v$. Putting everything together, we obtain the main lemma.

LEMMA 2.7. *If the algorithm chooses a minimum (negative) net-cost augmenting path at each step, then the minimum net-cost path at any time step is a downward path starting from the root.*

2.2 Algorithm description. By Lemmas 2.3 and 2.7, we proceed as follows: At each step, we find a minimum net-cost augmenting path of the form $r \rightsquigarrow u$ for some $u \in T$. If $\theta(r \rightsquigarrow u) > 0$, we stop. Otherwise, we augment the mass and flows as described above. We use a dynamic programming based approach to compute the described augmenting path and maintain some auxiliary information at each node and edge of T. For each edge $u \to v \in E$, we maintain the values of $K_1(u \to v), K(u \to v)$ and $\theta(u \to v)$. For each interior node v, define

$$\Theta(v) = \min\{0, \min_{u \in \mathsf{C}[v]} \{\Theta(u) + \theta(v \to u)\}\}$$

to be the net cost of the minimum net-cost augmenting path going down the tree starting from v. Define $\zeta(v)$, the *preferred child of* v, to be v if $\Theta(v) = 0$ and $\arg\min_{u \in \mathsf{C}[v]} \Theta(u) + \theta(v \to u)$ otherwise. Additionally, let $\zeta^*(v)$, the end of the minimum net cost down-tree augmenting path starting from v, be v if $\Theta(v) = 0$ and $\zeta^*(\zeta(v))$ otherwise.

Next, we define the bottleneck flow that can be sent along the minimum net-cost augmenting path starting from v through the preferred child, as follows. Let

$$\psi(v) = \min_{i \in K_1(v \to \mathsf{p}(v))} f_i(v \to \mathsf{p}(v))$$

be the smallest positive flow along the edge from v to its parent $\mathsf{p}(v)$. We denote this value as the *bottleneck flow value* along edge $v \to \mathsf{p}(v)$. Set the bottleneck $\delta(v)$ of a node v to be

$$\delta(v) = \begin{cases} \psi(v), & \Theta(v) = 0 \\ \min\{\delta(\zeta(v)), \psi(v)\}, & \text{otherwise.} \end{cases}$$

That is, $\delta(v)$ is the minimum of $\psi(v)$ along the most negative net-cost path starting from v. We denote this value as the *bottleneck flow value* along the minimum net cost down-tree augmenting path starting from v.

We now describe the algorithm. Compute initial flows $\boldsymbol{f}$ and set $\mu(r) = 1$ as above. By traversing the tree in a bottom-up manner, we compute $\Theta(v), \zeta(v)$ and $\zeta^*(v)$ for all nodes v of T. Repeat the following two steps until either $\mu(r) = 0$ for the root node r or $\Theta(r) \geq 0$, i.e., the minimum net-cost of an augmenting path starting from the root r is not negative. Suppose $\Theta(r) < 0$, $\mu(r) > 0$, and $\zeta^*(r) = z$. The minimum net-cost path $r \rightsquigarrow z$ can be retrieved by following $\zeta(\cdot)$ pointers.

1. **Augment:** Let $\delta^* = \min\{\delta(r), \mu(r)\}$. For each $u \to v$ along the minimum net cost augmenting path $r \rightsquigarrow z$, increment the down-flow $g(u \to v)$ by δ^*. Set $\mu(r) = \mu(r) - \delta^*$ and $\mu(\zeta^*(r)) = \mu(\zeta^*(r)) + \delta^*$.

2. **Update:** If $K_2(u \to v) \neq \varnothing$, then set $K_2(u \to v) = \varnothing$ and $K_1(u \to v) = K_1(u \to v) \cup K_2(u \to v)$. Additionally, for any $i \in K_1(v \to u)$ such that $f_i(v \to u) = \delta^*$, remove i from $K_1(v \to u)$ and add i to $K_2(u \to v)$. For each edge $u \to v$ on the path $r \rightsquigarrow \zeta^*(r)$, from the updated sets $K_j(u \to v)$ for $j = 1, 2$, recompute $k_j(u \to v) = |K_j(u \to v)|$ and $\theta(u \to v) = (k - 2k_1(v \to u)) \cdot c(u, v)$. Finally, update $\Theta(v), \zeta(v)$ and $\zeta^*(v)$ for all nodes v on $r \rightsquigarrow \zeta^*(r)$. We note that the value of $k_1(u \to \mathsf{p}(u))$ does not increase over time. So $\theta(\mathsf{p}(u) \to u)$ and $\Theta(u)$ are monotonically non-decreasing functions.

It is easily seen that each iteration of the algorithm takes $O(h)$ time. There are $O(nk)$ events since each iteration reduces $|K_1(u \to \mathsf{p}(u))|$ of at least one node u of T. Therefore, the overall runtime is $O(nkh)$, which is $\Omega(n^2 k)$ in the worst case. Next, we show that the runtime can be improved to $O(nk \min\{h, \log^2 n\})$ using the link-cut tree data structure [38].

A faster implementation. We compute a path decomposition $\mathcal{P}$ of T as follows. We call an edge $u \to v$, where $v \in \mathsf{C}[u]$, *solid* if $\Theta(u) < 0$ and $v = \zeta(u)$, and *dashed* otherwise. A maximal sequence of solid edges is a path P in the decomposition $\mathcal{P}$. There is at most one path in $\mathcal{P}$ that contains the root r of T, which we denote by P^* —it is the most negative net-cost path descending from r. We store each path P of $\mathcal{P}$ in a balanced tree $\mathcal{T}_P$ so that the lowest edge of P is the leftmost leaf of $\mathcal{T}_P$. For each edge $u \to \mathsf{p}(u)$, we implicitly maintain $g(\mathsf{p}(u) \to u), \psi(u)$ and $\theta(\mathsf{p}(u) \to u)$. Note that for any node $v \in P$, $\zeta^*(v)$ is the lowest node of P. By storing auxiliary information, we can also maintain $\Theta(v)$ and $\delta(v)$ for any node $v \in P$, as well as the lowest node, denoted by $\eta(v)$, that realizes $\delta(v)$ —the predecessor of v on P with the minimum value of ψ. For a non-root

node $u \in T$, let $\omega(u) = \Theta(u) + \theta(\mathsf{p}(u) \to u)$. For a node u with children x and y such that $\zeta(u) = x$, we define $\Delta\omega(u) = \omega(x) - \omega(y)$. We use $\Delta\omega(u)$ to determine when $\zeta(u)$ changes —$\zeta(u)$ switches from x to y when $\Delta\omega(u)$ becomes positive. We augment $\mathcal{T}_P$ so that we can maintain $\max_u \Delta\omega(u)$, where the maximum is taken over all nodes of P except the lowest one.

At each step, we perform operations on P^*. First, we increase $g(\mathsf{p}(u) \to u)$ by $\delta^* = \min\{\delta(r), \mu(r)\}$. Let $z = \eta(r)$. Next, we check whether $\psi(z) = g(\mathsf{p}(z) \to z)$, i.e., one of the i's needs to be moved from $K_1(z \to \mathsf{p}(z))$ to $K_2(z \to \mathsf{p}(z))$. This in turn updates $\psi(z)$ and $\delta(r)$. Change in $K_1(z \to \mathsf{p}(z))$ also increases $\theta(z \to \mathsf{p}(z))$ and $\Theta(v)$ for all successors v of z. We maintain this change at $O(\log n)$ nodes of the tree. Each of these operations can be performed in $O(\log n)$ time. We repeat this step until there is no edge $u \to \mathsf{p}(u)$ in P^* with $g(\mathsf{p}(u) \to u) = \psi(u)$. Next, we check if $\max_{z \in P^*} \Delta\omega(z) > 0$. Let z be the lowest node for which $\Delta\omega(z) = 0$. In this case, let $x = \zeta(z)$ and let y be the sibling of z. We make $x \to z$ as a dashed edge. If $\omega(y) < 0$, we make the edge $y \to z$ a solid edge. This in turn modifies P^* —we cut the edge $x \to z$ from P^*. The prefix of P^* ending at x becomes a new path in $\mathcal{P}$. Let P_1 be the suffix of P^* starting from z, and let P_2 be the path in $\mathcal{P}$ ending at y. Then we set $P^* = P_2 \circ y \to z \circ P_1$. Finally, if $\omega(y) \geq 0$, then we set $P^* = P_1$. Modifying P^* takes $O(\log n)$ time. We repeat this step as long as there is a node on P^* with $\Delta\omega(z) > 0$. We omit the details of the data structure from this version as they are similar to those in [38]. We now sketch the analysis of the overall runtime.

The preprocessing step takes $O(nk)$ time since one computes $O(k)$ values to store at each of the n vertices. In each iteration of the algorithm, we either decrease $\mu(r)$ to zero (happens at most once) or decrease the value of $k_1(u \to \mathsf{p}(u))$ for some u. There are at most $O(n)$ vertices, and for each vertex u we have $|K_1(u \to \mathsf{p}(u))| \leq k$. Therefore, there must be at most $O(nk)$ iterations of the algorithm. In each iteration, we augment some nonzero mass along the minimum net cost path P^*, recompute ψ, and increase the values of θ along the path P^*. Each of these operations can be done implicitly in $O(\log n)$ time (exact values can be recovered as a prefix sum or minimum along a path). Then using updated values of θ, we modify the path P^*. In the event of a change in preferred child, we link the new preferred child to the tree $\mathcal{T}_{P^*}$ in $O(\log n)$ time. Therefore, it suffices to bound the number of times a solid edge becomes a dashed edge.

We use the standard heavy-light decomposition [38]. Define an edge $v \to u$ such that $u \in \mathsf{C}[v]$ as *heavy* if $|T(u)| > \frac{1}{2}|T(v)|$ where $T(w)$ denotes the subtree of T rooted at w. Let any edge that is not heavy be called *light*. For every vertex v, at most one of its child edges is heavy by defintion. Let P^* be the minimum net cost path at a fixed iteration. Then observe there are at most $O(\log n)$ light edges on the path P^*. So at most $O(\log n)$ light edges change from solid to dashed in this iteration, since only edges of P^* can change from solid to dashed. Now suppose a heavy edge changes from solid to dashed during this iteration of the algorithm. Then either it is the first time this heavy edge changes away from solid to dashed, or in a prior iteration it became solid after the sibling light edge changed from solid to dashed. Therefore, after m circulations, the total number of times some heavy edge can become dashed is $O(m \log n) + (n-1)$. We conclude that the total number of times an edge becomes dashed is $O(nk \log n)$ since there are at most $m = O(nk)$ augmenting paths computed by the algorithm.

Putting everything together, we obtain the following.

LEMMA 2.8. *Let $T = (V, E)$ be a tree with $|V| = n$, height h, and edge costs $c\colon E \to \mathbb{R}_{\geq 0}$. Given a collection of k distributions $\mu_i\colon V \to [0,1]$ for $1 \leq i \leq k$, our algorithm computes an optimal barycenter $\mu^*\colon V \to [0,1]$ in $O(nk \min\{\log^2 n, h\})$ time.*

2.3 Computing dual weights. We now describe an algorithm for computing an optimal dual solution using the primal solution we computed. We begin by proving a useful property of an optimal dual solution.

LEMMA 2.9. *There exists an optimal solution $(\boldsymbol{\phi}, \lambda)$ to the dual barycenter problem where $\lambda = 0$.*

Proof. Suppose $(\boldsymbol{\phi}^* = \langle \phi_1^*, \ldots, \phi_k^* \rangle, \lambda^*)$ is an optimal solution to the dual problem. Let $\phi_i(u) = \phi_i^*(u) + \frac{\lambda^*}{k}$. Then,

$$\begin{aligned}\sum_i \sum_{u \in V} \phi_i(u)\mu_i(u) &= \sum_i \sum_{u \in V} \left(\phi_i^*(u) + \frac{\lambda^*}{k}\right) \cdot \mu_i(u) \\ &= \sum_i \sum_{u \in V} \phi_i^*(u)\mu_i(u) + \lambda^*,\end{aligned}$$

since $\sum_u \mu_i(u) = 1$. Moreover,

$$\phi_i(u) - \phi_i(v) = \left(\phi_i^*(u) + \frac{\lambda^*}{k}\right) - \left(\phi_i^*(v) + \frac{\lambda^*}{k}\right) = \phi_i^*(u) - \phi_i^*(v)$$

for all $(u, v) \in E$. Finally,

$$\sum_i \phi_i(v) = \sum_i \left(\phi_i^*(v) + \frac{\lambda^*}{k}\right) = \sum_i \phi_i^*(v) + \lambda^*$$

for all $v \in V$. We conclude that $(\boldsymbol{\phi} = \langle \phi_1, \ldots, \phi_k \rangle, 0)$ is a feasible solution with the same objective cost as the optimal solution $(\boldsymbol{\phi}^*, \lambda^*)$. ∎

Therefore it suffices to compute $\phi_i(u)$ for all $1 \le i \le k$ and for all $u \in T$ assuming $\lambda = 0$. Using Lemma 2.3, we compute a dual feasible solution that satisfies complementary slackness, i.e.,

(C1) if $f_i(u \to v) > 0$, then $\phi_i(u) - \phi_i(v) = c(u, v)$, and

(C2) if $\mu(u) > 0$, then $\sum_i \phi_i(u) = 0$.

For $i = 1, 2$, and an edge $u \to v$ of T, let $K_i^*(u \to v)$ and $k_i^*(u \to v)$ be the values of $K_i(u \to v)$ and $k_i(u \to v)$ with respect to the optimal flow computed above. Similarly define $\alpha^*(u \to v), \beta^*(u \to v)$ and $\theta^*(u \to v)$. Intuitively, one can view $\alpha^*(u \to v)$ as a required difference in potential from u to v to preserve the complementary slackness condition and $\beta^*(u \to v)$ as the amount of flexibility we have on the dual weights of the distributions which have no flow.

We choose an arbitrary vertex $z \in V$ with $\mu(z) > 0$ and reroot T so that z is the new root. Define $\Theta^*(u)$ based on this new root z. For a node $u \in T$, define $\Phi(u) = \sum_{i=1}^k \phi_i(u)$. We compute ϕ_i's recursively in a top-down manner. Set $\phi_i(z) = 0$ for all $1 \le i \le k$. Let u be a non-root node, and let v be its parent such that we have computed $\phi_i(v)$. We set

$$\phi_i(u) := \begin{cases} \phi_i(v) + c(u, v), & i \in K_1^*(u \to v), \\ \phi_i(v) - c(u, v), & i \in K_1^*(v \to u), \\ \phi_i(v) + \min\left\{c(u, v), \frac{-\Phi(v) + \Theta^*(u) - \alpha^*(u \to v)}{k_2^*(u \to v)}\right\}, & i \in K_2^*(u \to v). \end{cases} \tag{2.5}$$

The total time spent in computing the dual solution is $O(nk)$. Using the above definition of $\phi_i(u)$, we obtain the following:

LEMMA 2.10. *Given an edge $u \to v$ of T where u is a child of v after rerooting the tree at z,*

$$\Phi(u) = \min\{\Phi(v) + \theta^*(u \to v), \Theta^*(u)\}.$$

Proof.

$$\begin{aligned} \Phi(u) &= \sum_i \phi_i(u) \\ &= \sum_i \phi_i(v) + k_1^*(u \to v) \cdot c(u, v) - k_1^*(v \to u) \cdot c(u, v) \\ &\quad + k_2^*(u \to v) \cdot \min\left\{c(u, v), \frac{-\Phi(v) + \Theta^*(u) - \alpha^*(u \to v)}{k_2^*(u \to v)}\right\} \\ &= \Phi(v) + \alpha^*(u \to v) + \min\left\{\beta^*(u \to v), -\Phi(v) + \Theta^*(u) - \alpha^*(u \to v)\right\} \\ &= \min\{\Phi(v) + \theta^*(u \to v), \Theta^*(u)\}. \end{aligned} \tag{2.6}$$

The last equality follows because $\theta^*(u \to v) = \alpha^*(u \to v) + \beta^*(u \to v)$. ∎

We now prove feasibility and optimality of the constructed dual weights.

LEMMA 2.11. *The dual solution computed by the algorithm is feasible and optimal.*

Proof. We first prove the feasibility. By definition of Φ and Θ^*, $\Phi(u) \le \Theta^*(u) \le 0$. Hence, it suffices to prove that for any node u and its parent v, $|\phi_i(u) - \phi_i(v)| \le c(u,v)$.

For every $i \in K_1^*(u \to v) \cup K_1^*(v \to u)$, we observe that $|\phi_i(u) - \phi_i(v)| \le c(u,v)$ follows from the construction. Moreover, by the definition of ϕ_i,

$$\phi_i(u) - \phi_i(v) \le c(u,v)$$

for all $i \in K_2^*(u \to v)$. What remains is to show that $\phi_i(u) - \phi_i(v) \ge -c(u,v)$ for $i \in K_2^*(u \to v)$. Suppose

$$\frac{-\Phi(v) + \Theta^*(u) - \alpha^*(u \to v)}{k_2^*(u \to v)} < -c(u,v).$$

Then using definitions of θ and Θ,

$$\begin{aligned}
\Phi(v) &> -\alpha^*(u \to v) + k_2^*(u \to v) \cdot c(u,v) + \Theta^*(u) \\
&= -\alpha^*(u \to v) + \beta^*(u \to v) + \Theta^*(u) \\
&= \alpha^*(v \to u) + \beta^*(v \to u) + \Theta^*(u) \quad \text{(see (2.3))} \\
&= \theta^*(v \to u) + \Theta^*(u) \\
&\ge \Theta^*(v),
\end{aligned}$$

i.e. $\Phi(v) > \Theta^*(v)$, which contradicts Lemma 2.10. Hence, we conclude that $|\phi_i(u) - \phi_i(v)| \le c(u,v)$ for all $(u \to v)$ in T.

Next, we prove that the dual solution satisfies the complementary slackness conditions (C1) and (C2). If $f_i(u \to v) > 0$ then by construction of ϕ_i, $\phi_i(u) - \phi_i(v) = c(u,v)$. So we just need to prove (C2). First, note that for the (new) root z of T, $\mu^*(z) > 0$ and $\Phi(z) = 0$. Let u be a non-root node such that $\mu^*(u) > 0$. By Lemma 2.10 and definition of Θ^*, $\Phi(u) \le \Theta^*(u) \le 0$. Suppose $\Phi(u) < 0$. Let $u = v_0, v_1, v_2, \ldots, v_t = z$ be the path $P(u \rightsquigarrow z)$ from u to z in T. If $\Phi(v_i) = \Phi(v_{i+1}) + \theta^*(v_i \to v_{i+1})$ for all $i < t$, then

$$\Phi(u) = \sum_{i=1}^{t-1} \theta^*(v_i \to v_{i+1}) + \Phi(z) \ge \Theta^*(z) + \Phi(z) = 0.$$

Hence, $\Phi(u) = 0$ in this case. Therefore assume that $\Phi(v_i) < \Phi(v_{i+1}) + \theta^*(v_i \to v_{i+1})$ for some $i < t$. Let $j < t$ be the smallest index for which $\Phi(v_j) = \Theta^*(v_j)$. Let w be the descendant of v_j such that $\Theta^*(v_j) = \sum_{x \to y \in v_j \rightsquigarrow w} \theta^*(x \to y)$. Then

$$\begin{aligned}
\Phi(u) &= \sum_{i=1}^{j-1} \theta^*(v_i \to v_{i+1}) + \Phi(v_j) \\
&= \sum_{x \to y \in u \rightsquigarrow v_j} \theta^*(x \to y) + \Theta^*(v_j) \\
&= \sum_{x \to y \in u \rightsquigarrow v_j} \theta^*(x \to y) + \sum_{x \to y \in v_j \rightsquigarrow w} \theta^*(x \to y) \\
&= \sum_{x \to y \in u \rightsquigarrow w} \theta^*(x \to y).
\end{aligned}$$

If $\Phi(u) < 0$ then we have a path $u \rightsquigarrow w$ in T of negative net cost and $\mu^*(u) > 0$, which contradicts Lemma 2.3. Hence, (C2) holds.

Putting everything together, the dual solution is feasible and satisfies (C1) and (C2). Therefore by the theory of LP duality [12], the dual solution is optimal. ∎

Lemmas 2.8 and 2.11 together prove Theorem 1.2, the main result of this section.

3 Approximate Barycenter in Euclidean Space

We now describe the algorithm for constructing a $(1+\varepsilon)$-approximate barycenter in $\mathbb{R}^d$. We first give a high-level overview (Section 3.1), then describe the key steps in detail (Sections 3.2-3.5), and finally analyze its correctness and runtime (Section 3.6).

3.1 High-level overview. We begin by computing a small support set $Y \supseteq X$ (see Section 3.2) over which there exists a distribution $\mu\colon Y \to [0,1]$ satisfying $\Gamma(\mu) \leq (1+\varepsilon)\min_{\mu^*}\Gamma(\mu^*)$, where $\Gamma(\mu)$ denotes the cost of a barycenter distribution μ and the minimum is taken over all distributions μ^* with arbitrary support.

We then construct a randomized weighted graph $\mathcal{G} = (V, E)$ with $Y \subseteq V \subset \mathbb{R}^d$ and $|E| = O(n\varepsilon^{-2d}\log n)$ whose shortest path metric $\pi_{\mathcal{G}}$ is a $(1+\varepsilon)$-approximation of the Euclidean metric in expectation (see Section 3.4 below). Furthermore, $\mathcal{G}$ contains a randomly shifted binary tree $\mathcal{T}$ (which can be viewed as a variant of the BBD-tree [9], see Section 3.3) as a subgraph, and the tree metric $\pi_{\mathcal{T}}$ induced by $\mathcal{T}$ approximates $\pi_{\mathcal{G}}$ up to a multiplicative factor.

Recall that $X \subseteq Y \subseteq V$. For $1 \leq i \leq k$, let $\mu_i\colon V \to R_{\geq 0}$ be the input probability distribution with $\mu_i(p) = 0$ for all $p \in V \setminus X$. We compute a $(1+\varepsilon)$-approximate barycenter $\hat{\mu}$ for $\mu_1, \ldots, \mu_k$ with Y as support and $\pi_{\mathcal{G}}$ as the metric and using the flow formulation given in Section 1.1, as follows.

First, we call the Tree-Barycenter algorithm once on $\mathcal{T}$ to obtain an $O(\log n)$-approximate barycenter objective cost, i.e., we compute a flow satisfying $w^* \leq \tilde{g} \leq c\log n \cdot w^*$ where w^* is the optimal barycenter objective cost and c is a constant. Next, we perform a binary search in the range $\left[\frac{\tilde{g}}{c\log n}, \tilde{g}\right]$ to compute a $(1+\varepsilon)$-approximate barycenter. At each step of binary search, we have an interval $[g^-, g^+]$ and flow functions $\boldsymbol{f}$ with $¢(\boldsymbol{f}) \leq (1+\frac{\varepsilon}{3})g^+$ and $g^- \leq w^*$. Therefore if $g^+ \leq (1+\frac{\varepsilon}{3})g^-$, we return the probability distribution $\hat{\mu}$ associated with $\boldsymbol{f}$. So assume that $g^+ > (1+\frac{\varepsilon}{3})g^-$. We set $g = \frac{g^+ + g^-}{2}$ and use a MWU-based algorithm (see Section 3.5) that either computes a collection $\boldsymbol{f} = \langle f_1, \ldots, f_k\rangle$ of flows $f_i\colon E \to \mathbb{R}$ with $¢(\boldsymbol{f}) \leq (1+\frac{\varepsilon}{3})g$ or a set of dual weights $\boldsymbol{\phi} = \langle \phi_1, \ldots, \phi_k\rangle$ that certify $g < w^*$. We shrink the interval to $[g^-, g]$ in the former case and to $[g, g^+]$ in the latter case, and continue until the binary search interval has size at most $\frac{\varepsilon}{3}g^-$. Let $\hat{\mu}$ be the probability distribution returned by this procedure. By Lemma 3.3, $\hat{\mu}$ is a $(1+\varepsilon)$-approximate barycenter of $\mu_1, \ldots, \mu_k$ under the Euclidean metric in expectation.

3.2 Small candidate support set. For $1 \leq i \leq k$, let μ_i be the input distribution with finite support $X_i \subset \mathbb{R}^d$. Set $X = \bigcup_{i=1}^k X_i$ and $n = |X|$. We construct the desired set $Y \supseteq X$ as follows.

For any set $A \subseteq \mathbb{R}^d$, we define the *diameter* of A as $\mathrm{diam}(A) := \max_{x,y\in A}||x-y||$. Let $X \subset \mathbb{R}^d$ be a set of n points. Then given any arbitrary $\varepsilon > 0$, an *ε-well separated pair decomposition (WSPD)* is a set W of pairs of subsets $A, B \subseteq X$ satisfying:

1. $\max\{\mathrm{diam}(A), \mathrm{diam}(B)\} \leq \varepsilon \cdot \min_{a\in A, b\in B}||a-b||$ for every $(A,B) \in W$, and
2. for every $x \neq y \in X$, there exists some $(A,B) \in W$ such that $x \in A, y \in B$.

The first condition guarantees that each pair is *ε-well separated*, while the second condition guarantees the pairs of sets cover all possible pairs of points in the original point set. We construct an $(\frac{\varepsilon}{4})$-WSPD $W = \{(A_1,B_1), \ldots, (A_m,B_m)\}$ of size $m = O(n\varepsilon^{-d})$ in $O(n\varepsilon^{-d} + n\log n)$ time using the algorithm by Callahan and Kosaraju [14]. Choose representative points $a_i \in A_i$ and $b_i \in B_i$ arbitrarily for each pair $(A_i, B_i) \in W$.

Let β_0 and β_1 be some fixed constants. Given a length $\ell > 0$, define the grid $\mathbb{G}_\ell$ to be the uniform grid in $\mathbb{R}^d$ of side length ℓ. Set $t = \beta_1\log(\varepsilon^{-1})$. For $1 \leq i \leq m$ and for $0 \leq j \leq t$, let $\delta_{ij} = 2^j\frac{\beta_0\varepsilon}{2\sqrt{d}}\|a_i - b_i\|$. Let $\mathcal{C}_{ij}$ denote the set of hypercubes of the grid $\mathbb{G}_{\varepsilon\delta_j}$ intersecting $D(a_i,\ \delta_j) \cup D(b_i,\ \delta_j)$. Set $\mathcal{C} = \bigcup_{i=1}^m\bigcup_{j=0}^t \mathcal{C}_{ij}$. For a grid cell $\square$, let $c_\square$ be its center. We set Y as $Y = X \cup \{c_\square : \square \in \mathcal{C}\}$.

Lemma 3.1. *Let $\mu^*\colon \mathbb{R}^d \to [0,1]$ be an arbitrary probability distribution with finite support. Then there exists a distribution $\mu\colon Y \to [0,1]$ satisfying $\Gamma(\mu) \leq (1+\varepsilon)\Gamma(\mu^*)$.*

Proof. For a finite set $S \subset \mathbb{R}^d$, let $\chi(S) \in \mathbb{R}^d$ denote the 1-median of S, i.e., $\chi(S) = \arg\min_{x\in\mathbb{R}^d}\sum_{p\in S}||x-p||$. As shown in [6], we can assume the support of optimal barycenter μ^* is a subset of the 1-medians of all k-tuples

in $\prod_{i=1}^k X_i$:

$$\mathrm{supp}(\mu^*) \subseteq \left\{ \chi(S) : S \in \prod_{i=1}^k X_i \right\}.$$

Let $Z = \mathrm{supp}(\mu^*)$ be the finite support of μ^* for convenience. Additionally, let $f_i^* \colon X_i \times Z \to [0,1]$ be optimal transport plans from μ_i to μ^* for $1 \le i \le k$.

For each $z \in Z$, by the definition of the transport plan f_i^*, we know for every $1 \le i \le k$ there must exist some $x_i \in X_i$ such that $f_i^*(x_i, z) > 0$. Suppose, for some arbitrary $z \in Z$, we have $S = (x_1, \ldots, x_k) \in \prod_{i=1}^k X_i$ satisfies $f_i^*(x_i, z) > 0$ for all $1 \le i \le k$. With some slight abuse of notation, we will write $\boldsymbol{f}^*(S, z) = \min_i f_i^*(x_i, z)$. We can also assume if $\boldsymbol{f}^*(S, z) > 0$ then $z = \chi(S)$. By construction of of the collection of hypercubes $\mathcal{C}$, which covers a $\frac{1}{\varepsilon}\mathrm{diam}(\cup_i X_i)$ diameter ball about $\cup_i X_i$, we note that every $z \in Z$ is contained in some $\square \in \mathcal{C}$. It then suffices to prove that reassigning the mass $\mu^*(z)$ to the center $c_\square$ of the hypercube $\square$ in $\mathcal{C}$ containing z will incur an ε-relative error.

For an arbitrary $S = (x_1, \ldots, x_k) \in \prod_{i=1}^k X_i$ where $\boldsymbol{f}^*(S, \chi(S)) > 0$, define $\underline{i}(S) = \arg\min_i ||x_i - \chi(S)||$ to be the index of the point of S closest to $\chi(S)$. We drop the dependence on S when S is clear from context. Observe that by construction of each $\mathcal{C}_{ij}$, for any $j \neq \underline{i}(S)$,

$$||\chi(S) - c_\square|| \le \lambda\varepsilon ||x_j - \chi(S)||$$

for some fixed constant $\lambda > 0$ depending on β_0, β_1 and d. We then charge the additive error for $x_{\underline{i}}$ to the other indices $j \neq \underline{i}$ to get the desired result, assuming $k \ge 2$. Observe that

$$\begin{aligned}
\sum_{i=1}^k ||x_i - c_\square|| &\le \sum_{i=1}^k ||x_i - \chi(S)|| + ||\chi(S) - c_\square|| = \left(\sum_{i=1}^k ||x_i - \chi(S)|| \right) + k||\chi(S) - c_\square|| \\
&= \left(\sum_{i=1}^k ||x_i - \chi(S)|| \right) + \left(1 + \frac{1}{k-1} \right) \sum_{j \neq \underline{i}(S)} ||\chi(S) - c_\square|| \\
&\le \left(\sum_{i=1}^k ||x_i - \chi(S)|| \right) + 2\lambda\varepsilon \sum_{j \neq \underline{i}(S)} ||x_j - \chi(S)|| \le (1 + 2\lambda\varepsilon) \sum_{i=1}^k ||x_j - \chi(S)||.
\end{aligned}$$

We have therefore concluded that for any $S = (x_1, \ldots, x_k) \in \prod_{i=1}^k X_i$, there exists a cell $\square \in \mathcal{C}$ (in particular the cell containing $\chi(S)$) satisfying

$$\sum_{i=1}^k ||x_i - c_\square|| \le (1 + O(\varepsilon)) \sum_{i=1}^k ||x_j - \chi(S)||.$$

□

3.3 Randomized binary tree. We follow a construction similar to Fox and Lu [21]. Suppose we are given a d-dimensional axis-aligned box $B = \prod_{i=1}^d [a_i, b_i]$ and a finite set $Y \subset \mathbb{R}^d$ with $|Y| = n$ and spread $\Delta = \frac{\max_{a,b \in Y} \|a-b\|}{\min_{a,b \in Y} \|a-b\|}$. Let $\ell_i(B) = b_i - a_i$ be the length of B in dimension i, $\underline{\ell}(B) = \min_i \ell_i$ and $\overline{\ell}(B) = \max_i \ell_i$. Given a fixed dimension i and point $y \in B \cap Y$, define the *moat* of y in dimension i to be the interval $\left[y_i - \frac{\underline{\ell}(B)}{n^3}, y_i + \frac{\underline{\ell}(B)}{n^3} \right]$. A value $v_i \in [0, \ell_i(B)]$ is called *safe* with respect to the set Y and box B if v does not lie within any moats of $B \cap Y$, i.e., $|a_i + v_i - y_i| \ge \frac{\underline{\ell}(B)}{n^3}$ for all $y \in Y \cap B$.

We construct a *randomized binary tree with moats* $\mathcal{T} = T(Y)$ on Y, given $\varepsilon > 0$. Assume $Y \subseteq [-\Delta, \Delta]^d$ without loss of generality. Let $\square^* = [-2\Delta, 2\Delta]^d$ be the root cell of the tree. Add $c_{\square^*}$ as a vertex to $\mathcal{T}$. For each cell B of $\mathcal{T}$, we perform the following.

Define $\mathsf{A}[B]$ to be the smallest ancestor cell of B such that $\overline{\ell}(B) \le \frac{\varepsilon}{144 d \log n} \cdot \underline{\ell}(\mathsf{A}[B])$ if it exists, i.e., if B is not too close to the root cell. If $\mathsf{A}[B]$ exists and $|\mathsf{A}[B] \cap Y| = 1$, we create an edge from the unique leaf element y of $\mathsf{A}[B] \cap Y$ to c_B. Furthermore, if $0 < \mathrm{diam}(B \cap Y) \le \frac{\underline{\ell}(B)}{n^4}$, we recursively construct a tree with moats $\mathcal{T}_{B \cap Y}$ on $B \cap Y$. Let r be the root of $\mathcal{T}_{B \cap Y}$. Create an edge from r to c_B with cost $\|r - c_B\|$.

In the event that neither of these conditions are met, we split B into child cells. Let $i^* \in \arg\max_i \ell_i$. Then choose a safe value $v_{i^*} \in \left[\frac{\overline{\ell}(B)}{3}, \frac{2\overline{\ell}(B)}{3}\right]$ with respect to Y uniformly at random. We use the data structure of [21] to check whether v_{i^*} is safe in $O(\log n)$ time. Define the children of B as $\mathsf{C}[B] = \{B' \in \{B_1, B_2\} : B' \cap Y \neq \varnothing\}$, where $B_1 = \{x \in B : x_i \leq a_{i^*} + v_{i^*}\}$ and $B_2 = \{x \in B : x_i \geq a_{i^*} + v_{i^*}\}$. For each $B' \in \mathsf{C}[B]$, add $c_{B'}$ to $\mathcal{T}$ along with an edge $(c_{B'}, c_B)$ of cost $\|c_{B'} - c_B\|$. We recurse on each $B' \in \mathsf{C}[B]$.

This concludes the construction of the randomized binary with moats. Let $\pi_{\mathcal{T}}$ denote the shortest path distance on the tree $\mathcal{T} = T(Y)$. As shown in [1, 21], this construction gives $O(\log n)$ distortion in expectation.

LEMMA 3.2. *The randomized binary tree $\mathcal{T}$ has size $|\mathcal{T}| = O(n \log n)$ and can be constructed in $O(n \log^3 n)$ time with high probability. Furthermore, for any two points $x, y \in Y$, we have $\mathbb{E}[\pi_{\mathcal{T}}(x, y)] \leq O(\log n) \cdot \|x - y\|$.*

3.4 Randomized graph construction. First, construct a randomized binary tree with moats $\mathcal{T} = (V, E_V)$ on Y as described above. Let $\pi_{\mathcal{T}} \colon V \times V \to \mathbb{R}_{\geq 0}$ be the tree metric induced by $\mathcal{T}$. The weighted graph $\mathcal{G}$ will be formed by adding extra edges to $\mathcal{T}$ to guarantee pairwise (Euclidean) distances between points of Y are well approximated. Call the edges E_V of $\mathcal{T}$ the *vertical edges* or *tree edges* of $\mathcal{G}$.

We add the following edges to $\mathcal{T}$ to form $\mathcal{G}$. For each cell B in $\mathcal{T}$, define the *subcells* of B, written as $\mathsf{S}[B]$, to be the set of all descendants B'' of B in $\mathcal{T}$ where $\mathsf{A}[B''] = B$ and no ancestor B' of B'' satisfies $\mathsf{A}[B'] = B$. Then by definition of $\mathsf{A}[B'']$, we have that $\overline{\ell}(B'') \leq \frac{\varepsilon}{144 d \log n} \cdot \underline{\ell}(B)$. We construct a $\frac{\varepsilon}{\sqrt{d}}$-well separated pair decomposition [14] on the centers of subcells in $\mathsf{S}[B]$ and add an edge for each pair in the following way. Add an edge between c_{B_1} and c_{B_2} with cost $\|c_{B_1} - c_{B_2}\|$ for every pair $B_1, B_2 \in \mathsf{S}[B]$ where $\|c_{B_1} - c_{B_2}\| \leq \frac{3}{\sqrt{d}\varepsilon} \max_i \operatorname{diam}(B_i)$. Also add an edge between $c_{B'_1}$ and $c_{B'_2}$ with cost $\|c_{B'_1} - c_{B'_2}\|$ for every maximal pair of $\frac{\varepsilon}{\sqrt{d}}$-well separated descendants B'_1, B'_2 of B. Let E_B denote the set of edges added in this process on pairs of cells between B and $\mathsf{S}[B]$ in the tree $\mathcal{T}$. We refer to $E_H = \bigcup_{B \in \mathcal{T}} E_B$ as the *horizontal edges* or *shortcut edges* of $\mathcal{G}$. This concludes the construction of $\mathcal{G}$.

Let $\pi_{\mathcal{G}} \colon V \times V \to \mathbb{R}_{\geq 0}$ be the shortest path metric of $\mathcal{G}$. By Lemma 3.2, $|V| = O(n\varepsilon^{-d} \log n)$. It can be shown that $|E_H| = O(n\varepsilon^{-2d} \log n)$ since E_H includes a subset of pairs of cells in the WSPD construction [14] plus up to $O(\varepsilon^{-d})$ edges for each cell B. We emphasize $\overline{\ell}(B)/\underline{\ell}(B) \leq 3$ for any cell $B \in \mathcal{T}$ since we always divide B along its largest dimension. Thus $\mathcal{G}$ has $O(n\varepsilon^{-d} \log \Delta)$ vertices and $O(n\varepsilon^{-2d} \log \Delta)$ edges. The following property of $\mathcal{G}$, taken from [2, 21, 25], is crucial for our algorithm.

LEMMA 3.3. *For any pair of points $a, b \in Y$,*

(i) $\pi_{\mathcal{G}}(a, b) \geq ||a - b||$, *and*

(ii) $\mathbb{E}[\pi_{\mathcal{G}}(a, b)] \leq (1 + \varepsilon)||a - b||$.

Furthermore, for any edge $e = (u, v) \in E$,

(iii) $\pi_{\mathcal{T}}(u, v) \leq O\left(\frac{\log n}{\varepsilon}\right) \cdot \pi_{\mathcal{G}}(u, v)$.

Remark: We note that $\mathcal{G}$ is basically a $(1 + \varepsilon)$-spanner of Y, i.e. $\pi_{\mathcal{G}}$ approximates Euclidean distances up to a $(1 + \varepsilon)$-factor. Instead of using a deterministic algorithm for constructing a spanner (e.g. based on WSPD [14]), we use this randomized construction to ensure (iii) of Lemma 3.3, which will be critical for the analysis of our algorithm.

3.5 Multiplicative weight update procedure. Let $\mathcal{G} = (V, E)$ be the graph constructed above. Recall that $|V| = O(n\varepsilon^{-d} \log n)$ and $|E| = O(n\varepsilon^{-2d} \log n)$. We replace each undirected edge $(u, v) \in E$ with the directed edges $u \to v$ and $v \to u$ for convenience; their costs are $||u - v||$. Assume we have the algorithm TREE-BARYCENTER, which, given the tree $\mathcal{T} = (V, E_{\mathcal{T}})$ and distributions $\overline{\mu}_i \colon V \to \mathbb{R}$ satisfying $\sum_{v \in V} \overline{\mu}_i(v) = 1$ for all i, computes a primal solution $(\overline{\mu}, \overline{\boldsymbol{f}})$ and dual solution $\overline{\boldsymbol{\phi}}$ satisfying:

(a.) $\overline{\phi}_i(u) - \overline{\phi}_i(v) \leq \rho \cdot ||u - v||$ for all $u \to v \in E$, where $\rho = \frac{c_1 \log n}{\varepsilon}$,

(b.) the pair $(\overline{\mu}, \overline{\boldsymbol{f}})$ is a feasible primal solution on input $\overline{\mu}_i$, and

(c.) $\mathfrak{c}(\overline{\boldsymbol{f}}) \leq \sum_i \sum_{v \in V} \overline{\phi}_i(v) \overline{\mu}_i(v)$.

Let $\tau = \lceil 8\varepsilon^{-2}\rho^2 \log(k|E|)\rceil$. The algorithm will run in at most τ iterations. During each iteration, we maintain a (possibly infeasible) *guess* $\boldsymbol{f}^t$ of the optimal primal solution with $¢(\boldsymbol{f}^t) \le g$. Initially, assign

$$f_i^0(u \to v) = \frac{g}{k \cdot ||u - v|| \cdot |E|}$$

for every directed edge $u \to v \in E$ and every distribution i. Then, for each iteration $1 \le t \le \tau$, define

$$\overline{\mu}_i^t(u) = \mu_i(u) - \sum_{v:(u,v)\in E} \left(f_i^{t-1}(u \to v) - f_i^{t-1}(v \to u)\right)$$

over all $u \in V$. The values of $\overline{\mu}_i^t$ denote the infeasibility of the current primal solution guess f_i^{t-1}. Run Tree-Barycenter with input $\overline{\mu}_i^t$ to obtain primal variables $(\overline{\mu}^t, \overline{\boldsymbol{f}}^t)$ and dual variables $\boldsymbol{\phi}^t$ that satisfy Lemma 3.4. We note that $\sum_u \overline{\mu}_i^t(u) = \sum_u \mu_i(u)$ for any i, and therefore the barycenter $\overline{\mu}^t$ by itself is a feasible barycenter distribution and each flow $f_i^{t-1} + \overline{f}_i^t$ is a feasible flow that transports μ_i to μ. If $¢(\boldsymbol{f}^{t-1} + \overline{\boldsymbol{f}}^t) \le (1+\varepsilon)g$, then we have found a feasible flow $\boldsymbol{f}^{t-1} + \overline{\boldsymbol{f}}^t$ and barycenter $\overline{\mu}^t$ with cost at most $(1+\varepsilon)g$. Return this primal pair $(\overline{\mu}^t, \boldsymbol{f}^{t-1} + \overline{\boldsymbol{f}}^t)$.

Otherwise, $¢(\overline{\boldsymbol{f}}^t) > \varepsilon g$ and there is some non-trivially large barycenter mass which is not yet assigned naturally by the maintained guess $\boldsymbol{f}^{t-1}$. We update the guessed flow on each edge $(u \to v)$ using the computed dual solution. Define the *slack* of the dual solution along edge $u \to v$ as

$$s_i^t(u \to v) = \frac{\overline{\phi}_i^t(u) - \overline{\phi}_i^t(v)}{||u - v||}.$$

Then for all $u \to v \in E$ and for all $1 \le i \le k$, set

$$f_i^t(u \to v) \leftarrow \exp\left(\frac{\varepsilon}{2\rho^2} \cdot s_i^t(u \to v)\right) \cdot f_i^{t-1}(u \to v).$$

We emphasize that flow along an edge is increasing if the slack is large. After this multiplicative update, we rescale f_i^t by $\frac{g}{¢(\boldsymbol{f}^t)}$ to maintain $¢(\boldsymbol{f}^t) \le g$. If the algorithm does not terminate within τ rounds, we conclude that $\frac{1}{\tau}\sum_i \sum_{t=1}^{\tau} \overline{\phi}_i^t$ certifies that $g \le w^*$. This concludes the description of the algorithm for computing $(1+\varepsilon)$-approximate barycenters in Euclidean space.

3.6 Analysis. Lemma 3.1 implies that there exists a $(1+\varepsilon)$-approximate barycenter with support of Y, and Lemma 3.3 states $\mathbb{E}[\pi_{\mathcal{G}}(a,b)] \le (1+\varepsilon)\|a-b\|$ for all $a, b \in Y$. Therefore, the optimal solution to the barycenter flow problem on graph $\mathcal{G}$ is a $(1+\varepsilon)$-approximation of the Wasserstein barycenter problem in expectation.

Set $\rho = \max_{u,v\in V} \frac{\pi_{\mathcal{T}}(u,v)}{\pi_{\mathcal{G}}(u,v)}$. By Lemma 3.3, $\rho = O(\varepsilon^{-1}\log n)$. The following lemma follows from Lemma 2.11, the fact that $\mathcal{T}$ is a subgraph of $\mathcal{G}$, and the strong duality of linear programs.

Lemma 3.4. *Given $\boldsymbol{\mu} = \langle \mu_1, \ldots, \mu_k\rangle$, the Tree-Barycenter algorithm outputs a flow family $\boldsymbol{f} = \langle f_1, \ldots, f_k\rangle$, distribution μ and dual weights $\boldsymbol{\phi} = \langle \phi_1, \ldots, \phi_k\rangle$ that satisfy the following desired conditions of the oracle from Section 3.5:*

(T1.) $\phi_i(u) - \phi_i(v) \le \rho \cdot \|u - v\|$ *for all* $u \to v \in E$,

(T2.) $(\mu, \boldsymbol{f})$ *is a feasible solution to the barycenter problem, and*

(T3.) $¢(\boldsymbol{f}) \le \sum_i \sum_{v\in V} \mu_i(v)\phi_i(v)$.

Then using the same analysis as in [41], we can prove the following lemma.

Lemma 3.5. *Given a guess g of the barycenter objective value on the graph $\mathcal{G}$ and an algorithm Tree-Barycenter which computes a primal-dual pair satisfying conditions (T1)-(T3) in $Q(n)$ time, the multiplicative weight update algorithm either returns a family of flows $\boldsymbol{f} = \langle f_1, \ldots, f_k\rangle$ satisfying $¢(\boldsymbol{f}) \le (1+\varepsilon)g$ or dual weights $\boldsymbol{\phi} = \langle \phi_1, \ldots, \phi_k\rangle$ certifying $g \le w^*$ in $O((Q(n) + k|E|)\frac{\rho^2 \log k|E|}{\varepsilon^2})$ time.*

Plugging in $\rho = O(\varepsilon^{-1}\log n)$, $|E| = O(n\varepsilon^{-2d}\log n)$, and $Q(n) = O(nk\varepsilon^{-d}\log^3 n)$, Lemma 3.5 implies that each step of the binary search takes $O(nk\varepsilon^{-2d-4}\log^5 n \log(nk))$ time. Since the binary search takes $O(\log\log n)$ steps, the overall runtime is $O(nk\varepsilon^{-2d-4}\log^5 n \log(nk)\log\log n)$. This proves Theorem 1.1.

4 Conclusion

In this work, we presented the first $(1+\varepsilon)$-approximation algorithm for the Wasserstein barycenter problem for $p = 1$. Our algorithm exploits the geometry of distributions in Euclidean space to push the runtime to near-linear. In the process, we also presented the first near-linear time exact algorithm for the Wasserstein barycenter problem on trees. We leave a few open problems as future work.

- First, no known $(1+\varepsilon)$-relative approximation algorithm for the 2-Wasserstein barycenter problem exists. Is it possible to extend existing subquadratic time approximation algorithms for 2-Wasserstein distances to the barycenter problem?
- Second, our algorithm incurs exponential dependence on the dimension. Can one compute a $(1+\varepsilon)$-approximate barycenter in $n^{2-\delta}(d\varepsilon^{-1})^{O(1)}$ time for some $\delta > 0$?
- Finally, the existing polynomial-time algorithms for Wasserstein barycenters in constant dimension are only weakly polynomial time. This raises the question of whether there exists a strongly polynomial time algorithm which computes an exact Wasserstein barycenter in fixed dimensions.

Acknowledgement

Work by P.A. and K.Y. was supported by NSF grants CCF-22-23870 and IIS-24-02823, and by a US-Israel Binational Science Foundation Grant 2022131. Work by S.R. and P.S. was supported by NSF grant CCF-2223871. We thank the anonymous reviewers for their useful comments.

References

[1] P. K. Agarwal, K. Fox, D. Panigrahi, K. R. Varadarajan, and A. Xiao. Faster algorithms for the geometric transportation problem. In *Proc. 33rd International Symposium on Computational Geometry*, pages 7:1–7:16, 2017.
[2] P. K. Agarwal, S. Raghvendra, P. Shirzadian, and K. Yao. Fast and accurate approximations of the optimal transport in semi-discrete and discrete settings. In *Proc. 35th Annual Symposium on Discrete Algorithms*, pages 4514–4529, 2024.
[3] J. Altschuler, J. Niles-Weed, and P. Rigollet. Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration. *Proc. 31st Annual Conference on Neural Information Processing Systems*, 2017.
[4] J. M. Altschuler and E. Boix-Adsera. Wasserstein barycenters can be computed in polynomial time in fixed dimension. *Journal of Machine Learning Research*, 22(44):1–19, 2021.
[5] J. M. Altschuler and E. Boix-Adsera. Wasserstein barycenters are NP-hard to compute. *SIAM Journal on Mathematics of Data Science*, 4(1):179–203, 2022.
[6] E. Anderes, S. Borgwardt, and J. Miller. Discrete Wasserstein barycenters: Optimal transport for discrete data. *Mathematical Methods of Operations Research*, 84:389–409, 2016.
[7] A. Andoni and H. Zhang. Sub-quadratic $(1+\epsilon)$-approximate euclidean spanners, with applications. In *Proc. 64th Annual Symposium on Foundations of Computer Science*, pages 98–112, 2023.
[8] S. Arya and T. Malamatos. Linear-size approximate Voronoi diagrams. In *Proc. 13th Annual Symposium on Discrete Algorithms*, pages 147–155, 2002.
[9] S. Arya and D. M. Mount. Approximate range searching. In *Proc. 11th Annual Symposium on Computational Geometry*, pages 172–181, 1995.
[10] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.
[11] J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré. Iterative Bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2):A1111–A1138, 2015.
[12] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific Belmont, MA, 1997.
[13] N. Bonneel, J. Rabin, G. Peyré, and H. Pfister. Sliced and radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51:22–45, 2015.
[14] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM*, 42(1):67–90, 1995.
[15] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. 34th Annual ACM Symposium on Theory of Computing*, pages 380–388, 2002.
[16] L. Chen, R. Kyng, Y. P. Liu, R. Peng, M. P. Gutenberg, and S. Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *Proc. 63rd Annual Symposium on Foundations of Computer Science*, pages 612–623, 2022.

[17] S. Claici, E. Chien, and J. Solomon. Stochastic Wasserstein barycenters. In *Proc. 35th International Conference on Machine Learning*, pages 999–1008, 2018.
[18] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Proc. 27th Annual Conference on Neural Information Processing Systems*, 2013.
[19] M. Cuturi and A. Doucet. Fast computation of Wasserstein barycenters. In *International conference on machine learning*, pages 685–693, 2014.
[20] P. Dvurechenskii, D. Dvinskikh, A. Gasnikov, C. Uribe, and A. Nedich. Decentralize and randomize: Faster algorithm for Wasserstein barycenters. *Proc. 32nd Annual Conference on Neural Information Processing Systems*, 2018.
[21] E. Fox and J. Lu. A deterministic near-linear time approximation scheme for geometric transportation. In *Proc. 64th Annual Symposium on Foundations of Computer Science*, pages 1301–1315, 2023.
[22] S. Guminov, P. Dvurechensky, N. Tupitsa, and A. Gasnikov. On a combination of alternating minimization and nesterov's momentum. In *Proc. 38th International Conference on Machine Learning*, pages 3886–3898, 2021.
[23] P. Indyk and N. Thaper. Fast image retrieval via embeddings. In *Proc. 3rd Internat. Workshop on Statistical and Comput. Theories of Vision*, 2003.
[24] A. Khamis, R. Tsuchida, M. Tarek, V. Rolland, and L. Petersson. Scalable optimal transport methods in machine learning: A contemporary survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20, 2024.
[25] A. B. Khesin, A. Nikolov, and D. Paramonov. Preconditioning for the geometric transportation problem. In *35th International Symposium on Computational Geometry*, pages 15:1–15:14, 2019.
[26] A. Kroshnin, N. Tupitsa, D. Dvinskikh, P. Dvurechensky, A. Gasnikov, and C. Uribe. On the complexity of approximating wasserstein barycenters. In *Proc. 36th International Conference on Machine Learning*, pages 3530–3540, 2019.
[27] N. Lahn, D. Mulchandani, and S. Raghvendra. A graph theoretic additive approximation of optimal transport. *Proc. 33rd Annual Conference on Neural Information Processing Systems*, pages 13813–13823, 2019.
[28] T. Le, V. Huynh, N. Ho, D. Phung, and M. Yamada. Tree-Wasserstein barycenter for large-scale multilevel clustering and scalable bayes. *arXiv preprint arXiv:1910.04483*, 2019.
[29] T. Le, M. Yamada, K. Fukumizu, and M. Cuturi. Tree-sliced variants of Wasserstein distances. *Proc. 33rd Annual Conference on Neural Information Processing Systems*, 2019.
[30] L. Li, A. Genevay, M. Yurochkin, and J. M. Solomon. Continuous regularized Wasserstein barycenters. *Proc. 34th Annual Conference on Neural Information Processing Systems*, pages 17755–17765, 2020.
[31] T. Lin, N. Ho, X. Chen, M. Cuturi, and M. Jordan. Fixed-support Wasserstein barycenters: Computational hardness and fast algorithm. In *Proc. 34th Annual Conference on Neural Information Processing Systems*, pages 5368–5380, 2020.
[32] T. Lin, N. Ho, and M. I. Jordan. On the efficiency of entropic regularized algorithms for optimal transport. *Journal of Machine Learning Research*, 23(137):1–42, 2022.
[33] E. F. Montesuma, F. N. Mboula, and A. Souloumiac. Recent advances in optimal transport for machine learning. *arXiv preprint arXiv:2306.16156*, 2023.
[34] J. Orlin. A faster strongly polynomial minimum cost flow algorithm. In *Proc. 20th Annual Symposium on Theory of Computing*, pages 377–387, 1988.
[35] G. Peyré, M. Cuturi, et al. Computational optimal transport: With applications to data science. *Found. and Trends® in Machine Learning*, 11(5-6):355–607, 2019.
[36] G. Puccetti, L. Rüschendorf, and S. Vanduffel. On the computation of Wasserstein barycenters. *Journal of Multivariate Analysis*, 176:104581, 2020.
[37] J. Sherman. Generalized preconditioning and undirected minimum-cost flow. In *Proc. 28th Annual Symposium on Discrete Algorithms*, pages 772–780, 2017.
[38] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. In *Proc. 13th Annual Symposium on Theory of Computing*, pages 114–122, 1981.
[39] Y. Takezawa, R. Sato, Z. Kozareva, S. Ravi, and M. Yamada. Fixed support tree-sliced Wasserstein barycenter. In *Proc. 25th International Conference on Artificial Intelligence and Statistics*, pages 1120–1137, 2022.
[40] C. Villani. *Optimal Transport: Old and New*. Springer, 2009.
[41] G. Zuzic. A simple boosting framework for transshipment. In *Proc. 31st Annual European Symposium on Algorithms*, pages 104:1–104:14, 2023.