We prove that the tree construction of Fakcharoenphol, Rao, and Talwar [2] can be used to approximate snowflake metrics by trees with expected distortion bounded independently of the size of the metric space. The constant of distortion we derive depends linearly on the dimension of the metric space. We also present an algorithm for building a single tree whose cost is linear in the problem size.

A note on approximating snowflake metrics by trees

William Leeb
Technical Report YALEU/DCS/TR-1501
November 20, 2014

Dept. of Mathematics, Yale University, New Haven CT 06511

Approved for public release: distribution is unlimited.
Keywords: *tree metric, partition trees, tree approximation, snowflake metric, dimension of a metric space*

1 Introduction

A basic problem in the theory of metric spaces is to approximate an arbitrary metric $d(x, y)$ on a finite set X by a convex combination of dominating tree metrics [1]. In other words, we seek a random family of partition trees $\mathcal{T}$ on X, with tree metrics $d_{\mathcal{T}}(x, y)$, such that

$$d(x, y) \leq d_{\mathcal{T}}(x, y) \tag{1}$$

for all trees $\mathcal{T}$ in the collection, and

$$\mathbb{E}_{\mathcal{T}} d_{\mathcal{T}}(x, y) \leq K d(x, y) \tag{2}$$

for some constant $K > 0$. In general, one cannot hope for K to be smaller than $\mathcal{O}(\log |X|)$, as shown by the example of the n-cycle [5]. The paper [2] describes a randomized construction of partition trees that achieves this $\mathcal{O}(\log |X|)$ bound for any metric space X. If $|X|$ is large, however, a size $\mathcal{O}(\log |X|)$ distortion can be too big to be useful.

In Section 3, we make the following simple observation about the trees considered in [2]. For any $0 < \alpha < 1$, these trees can be used to approximate the *snowflake metric* $d(x, y)^\alpha$ with expected distortion $\mathcal{O}(\dim(X)/(1 - \alpha))$, where $\dim(X)$ is a version of the doubling dimension of X that we define in Section 2.

In Section 4 we give an algorithm for constructing the trees in [2] that has cost $\mathcal{O}(|X^2|)$. Though such an algorithm is alluded to in [2], we have not seen it described in the literature. We also point out that the naïve algorithm described in [2] and elsewhere can have arbitrarily high running time independent of $|X|$.

2 Background: dimension, snowflake metrics and trees

If (X, d) is a metric space, we define its *dimension* to be

$$\dim(X) = \sup_{x \in X, r \geq 0} \log_2 \left(\frac{V(x, 2r)}{V(x, r)} \right)$$

where $V(x, r)$ denotes the number of points contained in the closed ball of radius r around the point $x \in X$. The key fact about the quantity $\dim(X)$ is that it often does not depend on the number of points in X. For example, a uniform grid of points in d-dimensional Euclidean space will have dimension approximately equal to d, independent of the number of points in the grid.

Another basic idea we need is that of the *snowflake* metric. A snowflake metric $\rho(x, y)$ satisfies the property that $\rho(x, y)^p$ is also a metric for some $p > 1$. Put another way, if we start start with any metric $d(x, y)$, the new metric $\rho(x, y) = d(x, y)^\alpha$ is a snowflake metric whenever $0 < \alpha < 1$.

The snowflake metric $d(x, y)^\alpha$ is often better-behaved than the original metric $d(x, y)$. There are many manifestations of this phenomenon. In classical analysis, for instance, spaces of functions that are smooth with respect to the snowflake metric $|x - y|^\alpha$ can be easily characterized along with their duals when $\alpha < 1$, but not when $\alpha = 1$ ($|x - y|$

denotes Euclidean distance) [4]. Another example is Assouad's Theorem; any snowflake metric space subject to a doubling condition can be embedded into $\mathbb{R}^n$ for some n [3]. Such embeddings do not exist for arbitrary metric spaces; the Heisenberg group is a counterexample [6].

Finally, we introduce the definition of *tree metric* that we will be using in this paper. A *partition tree* $\mathcal{T}$ on a set X is a collection of subsets $F \subset X$, which we will call *folders*, with the following properties:

1. The set X itself is in $\mathcal{T}$;
2. All the singletons $\{x\}$ lie in $\mathcal{T}$; and
3. For any two folders F and F' in $\mathcal{T}$, either $F \subset F'$, $F' \subset F$, or F and F' are disjoint.

We assume that each folder $F \in \mathcal{T}$ has a weight $w(F)$ placed on it. We require that if $F \subsetneq F'$, then $w(F) < w(F')$; and that every singleton folder as weight zero, or $w(\{x\}) = 0$ for all $x \in X$. We then define the tree distance $d_{\mathcal{T}}(x, y)$ between x and y to be the weight of the smallest folder containing both x and y. It is easy to see that this is a distance.

3 Approximating snowflake distances by random trees

Let (X, d) be a finite metric space. Assume that the diameter of X (that is, the maximum distance between any two points) is 1 (note that this convention is not standard; many authors scale the metric to have minimum distance 1).

We now describe the random trees of [2]. Much of the notation we use here comes from that paper. There are two random objects that define each tree: a random permutation π of the points in X, and a random number $\beta \sim \text{Uniform}(1, 2)$. We require that π and β be drawn independently of each other. Define $\beta_l = 2^{-l}\beta$.

For each $x \in X$ and $l \geq 0$, let x_l^* be the first point in X (according to the permutation π) such that $d(x, x_l^*) \leq \beta_l$. We will say that "x has been *assigned* to x_l^* at level l."

We recursively define a partition tree on X as follows. The only level 0 folder in the tree is the entire set X, and its center is the first point on the list (since the diameter of X is 1, and $\beta_0 \geq 1$). For each folder F at level l, the subfolders of F (at level $l + 1$) are formed by grouping together the points in F that were assigned to the same point at level $l + 1$.

Consequently, for each folder F at level l, there is a point x_F such that all the points $x \in F$ were assigned to x_F at level l; in particular, $d(x, x_F) < \beta_l$. Therefore, for any two points x and x' in F, $d(x, x') < 2\beta_l \leq 2^{-l+2}$.

Each point in X therefore belongs to a single folder, one at each level of the tree. We define the tree distance between two points to be $d_{\mathcal{T}}(x, y) = 2^{-l+2}$, where l is the last level at which x and y are in the same folder. We will denote the diameter of a set $S \subset X$ under the distance induced by T as $\text{diam}_{\mathcal{T}}(S)$, and the diameter of S in the original metric d as $\text{diam}(S)$.

Note that if F is this folder for x and y, then as we have seen the diameter of F (in the metric d) is no more than $2\beta_l$; therefore, the tree distance $d_{\mathcal{T}}(x, y)$ is an upper bound on the distance $d(x, y)$, and so $\operatorname{diam}_{\mathcal{T}}(S) \geq \operatorname{diam}(S)$. Consequently, the same inequality holds for the snowflake metrics: $\operatorname{diam}_{\mathcal{T}}(S)^\alpha \geq \operatorname{diam}(S)^\alpha$.

We will prove the following theorem:

Theorem 1. *The tree construction of [2] produces a family of trees $\mathcal{T}$ such that for any $0 < \alpha < 1$ and any subset $S \subset X$,*

$$\mathbb{E}_{\mathcal{T}} \operatorname{diam}_{\mathcal{T}}(S)^\alpha \leq K \operatorname{diam}(S)^\alpha$$

where $K = \mathcal{O}(\dim(X)/(1-\alpha))$ as $\alpha \to 1^-$.

In fact, we can prove trivially that the distortion for snowflakes can never be worse than the distortion of the original metric, raised to the power α. This follows trivially from Jensen's inequality, as

$$\frac{\mathbb{E}_{\mathcal{T}} \operatorname{diam}_{\mathcal{T}}(S)^\alpha}{\operatorname{diam}(S)^\alpha} = \mathbb{E}_{\mathcal{T}} \left(\frac{\operatorname{diam}_{\mathcal{T}}(S)}{\operatorname{diam}(S)} \right)^\alpha \leq \left(\frac{\mathbb{E}_{\mathcal{T}} \operatorname{diam}_{\mathcal{T}}(S)}{\operatorname{diam}(S)} \right)^\alpha.$$

In particular, since $\alpha < 1$, the snowflake's distortion is smaller than that of the original metric. Of course, this simple calculation does not prove that the distortion for the snowflake is bounded independently of the size of X; this is the content of Theorem 1.

The remainder of this section will be devoted to proving Theorem 1. The basic outline of the proof is the same as in [2]; the proofs differ at the end to account for the snowflake, and there are minor adjustments throughout due to our analyzing an arbitrary set S rather than a pair of points ([2] considers the case $S = \{x, y\}$ only).

Define the integer $l^* \geq 0$ by

$$2^{-l^*-1} < \operatorname{diam}(S) \leq 2^{-l^*}.$$

Observe that if all points in S are in the same folder at level l, the diameter of S must be less than the diameter of their shared folder, which implies that $\operatorname{diam}(S) \leq 2^{-l+1}$. Therefore,

$$l \leq l^* + 2.$$

For brevity, denote by G_l the event that all points in S are assigned to the same point at level l. Then we have shown

$$\mathbb{E}_{\mathcal{T}} \operatorname{diam}_{\mathcal{T}}(S)^\alpha \leq \sum_{l=0}^{\infty} 2^{-(l-2)\alpha} \Pr[G_l \setminus G_{l+1}] \leq \sum_{l=0}^{l^*+2} 2^{-(l-2)\alpha} \Pr[G_{l+1}^c]. \tag{3}$$

We will prove an upper bound on $\Pr[G_{l+1}^c]$ that will give us the desired result.

We introduce some language and notation that we will use throughout the proof:

- For each $y \in X$, let w_y be any point in S that is closest to y.

- For each $y \in X$, let x_y be any point in S that is farthest away from y.
- For each $y \in X$, let L_y denote the number of points in X that are as close or closer to S than y.
- Say that a point $y \in X$ *splits* S *at level* l if y is the first point to which any point in S is assigned at level l, but not all points of S are assigned to y at level l.

We state the following lemma:

Lemma 1. *At most one point can split S at level l. Furthermore, if y splits S at level l, then w_y must be assigned to y and x_y must not be assigned to y at level l.*

Proof. The first claim is obvious. For the second claim, suppose that y splits S at level l. Suppose for contradiction that w_y were not assigned to y. There must be some point x that is assigned to y, or else it could not split S; consequently, $d(x, y) \leq \beta_l$. Since w_y is the closest point to y in S, $d(w_y, y) \leq d(x, y) \leq \beta_l$; so the point to which w_y is assigned at level l must be a point in X that precedes y; call it y'. But x is assigned to y, not y'; and therefore, since y' precedes y, y cannot be the point that splits S (it must be y' or an even earlier point); contradiction. Therefore, w_y is assigned to y.

Similarly, suppose for contradiction that y splits S at level l, and that x_y is assigned to y at level l; then $d(x_y, y) \leq \beta_l$. Since x_y is the point farthest away from y in S, it follows that every other point in S is within β_l of y as well. The only way that y could split S, therefore, is if there were some other point $y' \in X$ preceding y such that some point $w \in S$ is assigned to y'. But x_y is assigned to y, not to y', and since y' precedes y, y cannot split S; contradiction. Therefore, x_y cannot be assigned to y, and the proof is complete. □

We now begin to develop the upper bound on $\Pr[G^c_{l+1}]$. The next inequality is obvious from the definitions:

Lemma 2. *Let $l \geq 0$. Then*

$$Pr[G^c_{l+1}] \leq \sum_{y \in X} Pr[y \text{ splits } S \text{ at level } l+1]. \tag{4}$$

To bound $\Pr[G^c_{l+1}]$ we will bound the probabilities $\Pr[y$ splits S at level $l+1]$ that appear in (4). Recall that L_y denotes the number of points as close or closer to S than y.

Lemma 3. *For any point $y \in X$ and any $l \geq 0$,*

$$Pr[y \text{ splits } S \text{ at level } l+1] \leq \frac{1}{L_y} 2^{l+1} \operatorname{diam}(S).$$

Proof. Take any $y \in X$. In order for y to split S at level $l+1$, it is necessary that the following two events occur:

(A) $\beta_{l+1} \in [d(w_y, y), d(x_y, y))$;

(B) y appears before all other points in X within distance β_{l+1} of S.

The necessity of event A follows immediately from Lemma 1. Event B is necessary, for otherwise the point that appears before y would be the first point on the list to which some point of S is assigned, making it impossible for y to split S.

We claim that, conditional on any fixed value of β, the probability of event B can be bounded above by $1/L_y$. This follows because if y is within β_{l+1} of S, then so is any point that is closer to S than y; in order for y to split S, it must appear before all these other points. The probability that y appears first is no more than $1/L_y$, since all permutations of these L_y points is equally likely to occur.

Using the triangle inequality and the fact that $\beta \sim \mathrm{Uniform}(1, 2)$, the probability of event A can be upper bounded by

$$\Pr[A] \leq \frac{d(x_y, y) - d(w_y, y)}{2^{-l-1}} \leq 2^{l+1} d(w_y, x_y) \leq 2^{l+1} \operatorname{diam}(S).$$

Therefore

$$\Pr[y \text{ splits } S \text{ at level } l+1] \leq \Pr[A \cap B] \leq \Pr[A]\Pr[B|A] \leq \frac{1}{L_y} 2^{l+1} \operatorname{diam}(S)$$

as desired. $\square$

We now derive the upper bound on $\Pr[G_{l+1}^c]$ that, when plugged into (3), will yield Theorem 1.

Lemma 4. *Suppose $l \leq l^* - 2$. Then*

$$Pr[G_{l+1}^c] \leq \frac{6}{\ln 2} 2^l \operatorname{diam}(S) \dim(X).$$

Proof. If

$$d(w_y, y) \leq 2^{-l-2}$$

then

$$d(x_y, y) \leq d(x_y, w_y) + d(w_y, y) \leq 2^{-l^*} + 2^{-l-2} \leq 2^{-l-2} + 2^{-l-2} = 2^{-l-1} \leq \beta_{l+1}$$

and so x_y is within β_{l+1} of y, which implies that x_y is assigned to y at level $l+1$, and so, by Lemma 1, y does not split S at level $l+1$. So if $l \leq l^* - 3$, in order for y to split S at level $l+1$, it must be that $d(w_y, y) > 2^{-l-2}$. Lemma 1 also implies it is necessary that $d(w_y, y) \leq \beta_{l+1} \leq 2^{-l}$. Consequently, the only points in X that could possibly split S at level $l+1$ are those whose distance to S is no greater than 2^{-l} and strictly greater than 2^{-l-2}.

Now, list the points in X in order of their distance to S:

$$y_1, \dots, y_{|X|}.$$

Ties are broken arbitrarily; all that matters is that there are at least j points in X as close or closer to S as is y_j; in other words, $L_{y_j} \geq j$.

Let I be the number of points in X whose distance to S is less than or equal to 2^{-l-2}, and J the number of points whose distance to S is less than or equal to 2^{-l}. Then the points whose distance to S is no greater than 2^{-l} and strictly greater than 2^{-l-2} are

$$y_{I+1}, \dots, y_J$$

and hence these are the only points that could possibly split S at level $l+1$. Consequently, applying Lemmas 2 and 3 and the fact that $L_{y_j} \geq j$ we get

$$\begin{aligned} \Pr[G_{l+1}^c] &\leq \sum_{y \in X} \Pr[y \text{ splits } S \text{ at level } l+1] \leq \sum_{j=I+1}^{J} \Pr[y_j \text{ splits } S \text{ at level } l+1] \\ &\leq \sum_{j=I+1}^{J} \frac{1}{j} 2^{l+1} \operatorname{diam}(S) \leq \frac{1}{\ln(2)} 2^{l+1} \operatorname{diam}(S) \log_2(J/I). \end{aligned} \tag{5}$$

The result will follow if we can prove that $\log_2(J/I) \leq 3 \dim(X)$. To show this, take any $x \in S$. Observe that if y is within 2^{-l} of S, then $d(x, y) \leq d(x, w_y) + d(w_y, y) \leq 2^{-l^*} + 2^{-l} \leq 2^{-l+1}$ (since $l \leq l^* - 2$), and therefore $J \leq V(x, 2^{-l+1})$; also, $V(x, 2^{-(l+2)}) \leq I$ (if $y \in X$ is within $2^{-(l+2)}$ of x, then obviously y is within $2^{-(l+2)}$ of S). Therefore,

$$\log_2(J/I) \leq \log_2 \left(\frac{V(x, 2^{-(l-1)})}{V(x, 2^{-(l+2)})} \right) \leq 3 \dim(X)$$

as desired. $\square$

We can now combine (3) and Lemma 4 to prove Theorem 1. We have

$$\begin{aligned} \mathbb{E}_{\mathcal{T}} \operatorname{diam}_{\mathcal{T}}(S)^\alpha &\leq \sum_{l=0}^{l^*} 2^{-(l-2)\alpha} \Pr[G_{l+1}^c] = \left\{ \sum_{l=0}^{l^*-2} + \sum_{l=l^*-1}^{l^*} \right\} 2^{-(l-2)\alpha} \Pr[G_{l+1}^c] \\ &\leq \frac{6}{\ln 2} \operatorname{diam}(S) \dim(X) 4^\alpha \sum_{l=0}^{l^*-2} 2^{l(1-\alpha)} + 4^\alpha \sum_{l=l^*-1}^{l^*} 2^{-l\alpha} \\ &\leq \frac{6}{\ln 2} 2^{-l^*} \dim(X) 4^\alpha \frac{1}{2^{1-\alpha} - 1} 2^{(l^*-1)(1-\alpha)} + 4^\alpha (2^\alpha + 1) 2^{-l^*\alpha} \\ &\leq 4^\alpha \left(\frac{\dim(X)}{2^{1-\alpha} - 1} \frac{3 \cdot 2^{-\alpha}}{\ln 2} + 2^\alpha + 1 \right) 2^{-l^*\alpha} \\ &\leq 8^\alpha \left(\frac{\dim(X)}{2^{1-\alpha} - 1} \frac{3 \cdot 2^{-\alpha}}{\ln 2} + 2^\alpha + 1 \right) \operatorname{diam}(S)^\alpha. \end{aligned}$$

Consequently, we have shown that

$$\mathbb{E}_{\mathcal{T}} \operatorname{diam}_{\mathcal{T}}(S)^{\alpha} \leq K \operatorname{diam}(S)^{\alpha}$$

where $K = \mathcal{O}(\dim(X)/(1-\alpha))$ as $\alpha \to 1^{-}$, proving Theorem 1.

4 The algorithm for building a single tree

In this section we describe an explicit algorithm for constructing a single tree $\mathcal{T}$, given the permutation π and the parameter β. We will show that the algorithm has cost $\mathcal{O}(|X|^2)$, which is linear in the problem size. In [2], the authors state the existence of such an algorithm, though we have not encountered it in the literature.

A naïve construction of the trees from [2] may repeat the same folder multiple times at different levels. This will occur when all the points in a folder at level l are assigned to the same point at level $l+1$. Of course, the tree distance in this case will only be determined by the copy of this folder at the smallest level, so there is no need to include the redundant copies in the tree.

Furthermore, we note that any algorithm whose running time is to be controlled solely in terms of the size of X must avoid forming redundant folders. To see this, consider a metric space with three points, $X = \{x, y, z\}$. Suppose $d(x, y) = \epsilon$, $d(x, z) = 1$, and $d(y, z) = 1 - \epsilon$ for some $\epsilon < 1/2$. Suppose too that the permutation π places x first, y second, and z third. Then it is easy to see that if $0 < l < \log_2(\beta/\epsilon)$, then the level l partition consists of the two folders $F_1 = \{x, y\}$ and $F_2 = \{z\}$; in particular, there are at least $\log_2(\beta/\epsilon)$ many levels before the tree splits into singletons and the construction terminates. Consequently, if an algorithm performs operations level-wise, the running time on this example can be made arbitrarily large by taking $\epsilon \to 0$. An algorithm whose cost depends only on $|X|$ must automatically skip over redundant folders.

We present such an algorithm in this section. Throughout the remainder of this section, we will list the points in the order given by π as $x_1, \dots, x_{|X|}$. The following lemma will be useful.

Lemma 5. *Suppose x has been assigned to x_{k_0} at level l_0, and to x_{k_1} at level l_1. Then if $l_1 > l_0$, it must be that $k_1 \geq k_0$.*

Proof. Suppose $k_1 < k_0$, i.e. x_{k_1} occurs before x_{k_0} on the list. Since x is assumed to be assigned to x_{k_1} at level l_1, therefore $d(x, x_{k_1}) \leq \beta_{l_1} < \beta_{l_0}$. But then, since x_{k_1} precedes x_{k_0}, x would have been assigned to x_{k_1} at level l_0; this is a contradiction. $\square$

In other words, we never need to backtrack through the list when looking for the next point to which x is assigned.

Given any points x and y in X define

$$l(x, y) = \lfloor \log_2(\beta) - \log_2(d(x, y)) \rfloor.$$

Lemma 6. *Suppose that x has been assigned to x_k at level l. Then $l \leq l(x, x_k)$, and x will be assigned to x_k at all levels l' such that $l \leq l' \leq l(x, x_k)$.*

Proof. By definition, $l(x, x_k)$ is the largest integer such that $d(x, x_k) \leq 2^{-l(x,y)}\beta$. Since $d(x, x_k) \leq \beta_l = 2^{-l}\beta$, we must have $l \leq l(x, x_k)$. Now suppose that x gets assigned to x_j at level l', $l \leq l' \leq l(x, x_k)$. By Lemma 5, $j \geq k$, i.e. x_j does not occur before x_k in the list. On the other hand, $d(x, x_k) \leq 2^{-l(x,y)}\beta \leq 2^{-l'}\beta = \beta_{l'}$; so x will not be assigned to any point occurring after x_k at level l'. Consequently, x is assigned to x_k at level l'. □

In other words, if x is ever assigned to a point y, then $l(x, y)$ is the last level at which x is assigned to y.

We introduce some terminology. For every folder F on the tree, we will refer to:

- The *center* of F. This is the point x_k that all points in F were assigned to when they became members of F.
- The *level* of F. This is the minimum of the numbers $l(x, x_k)$ for $x \in F$, denoted l'. Then $2^{-l'+2}$ is an upper bound for the diameter of F.

Observe that if a folder has center x_k and level l', then by Lemma 6, $l' + 1$ is the first level at which the folder F can be split into subfolders. Consequently, when we are splitting F into its children we never need to consider any subfolders at levels less than l', since they will all be equal to F. Also, if $x \in F$ and $l(x, x_k) = l'$, by Lemma 5 the point x_j to which x is assigned at level $l' + 1$, is the first point on the list $x_{k+1}, x_{k+2}, \ldots$ such that $l(x, x_j) > l'$.

These observations yield the following algorithm for constructing the tree. Initialize the tree with the single folder X, with center point x_1. Recursively build folders as follows. Take any folder F whose children have not yet been added to the tree. Let x_k be its center and l' its level. Take those points $x \in F$ with $l(x, x_k) > l'$, if any exist. These points will remain assigned to x_k at level $l' + 1$. So one of the children of F will consist of all points with $l(x, x_k) > l'$, if there are any.

The points with $l(x, x_k) = l'$ are no longer assigned to x_k at level $l' + 1$. To find where they go, for each such point search through $x_{k+1}, x_{k+2}, \ldots$ until the first x_j is encountered with $l(x, x_j) > l(x, x_k)$. This x_j is the next point to which x is assigned. Therefore, the remaining children of F are formed by grouping together those points that have been advanced to the same point in this manner.

Of course, it could happen that the numbers $l(x, x_k)$ are equal for all $x \in F$, and that all $x \in F$ get advanced to the same point x_j after x_k. In this case, we can keep the identity of F intact, update its center to x_j, find its new level, and repeat the process.

We give a summary of the algorithm:

Algorithm for building $\mathcal{T}$

I. Initialize the tree with folder X and center x_1.

II. Take any non-singleton folder F with center x_k and level l' whose children are not on the tree

1. If possible, form a child F_0 of F consisting of points with $l(x, x_k) > l'$
2. Advance each remaining x to the first x_j, $j > k$, with $l(x, x_j) > l'$
3. There are two cases:
   i. If $F_0 = \emptyset$ and all points in F advanced to the same point x_j, simply make x_j the new center of F and update F's level $l' = \min_{x \in F} l(x, x_j)$
   ii. Otherwise, break $F \setminus F_0$ into children of F by grouping the points that advanced to the same x_j, and let l' be the level of F

Repeat step II until the children of every folder have been added to the tree.

We now analyze the cost of this algorithm. Observe that every time a folder is processed, the operations fall into two categories. First, there is the cost of advancing each point $x \in F$ to the next point to which it is assigned. However, once a point x is advanced to x_k, it is never necessary, when considering x, to look at any points preceding x_k in the list, by Lemma 5; so the most that all such advances can cost over the entire algorithm is $\mathcal{O}(|X|^2)$, since each point x sweeps over all the points in X exactly once.

Second, there are those whose costs are directly proportional to the number of points in F, such as the cost of computing $l(x, x_k)$ for each x, where x_k is the center of F. We will break these costs into two cases. The first is when the folder F ends up being broken apart into subfolders. Since this only happens once per folder, the total cost of all such operations can be bounded above by a constant times

$$\sum_{F \in \mathcal{T}} |F| \leq \sum_{F \in \mathcal{T}} |X| = \mathcal{O}(|X|^2)$$

since there are at most $2|X| - 1$ folders in the tree.

The second case is when F does not get broken into subfolders. This can only happen when every point in F is advanced to the same point (so the center of F changes, but F is not broken apart). This does not pose any additional costs, however, since we have already counted these costs when we computed the cost of all advances.

The total cost of the algorithm, therefore, is $\mathcal{O}(|X|^2)$.

References

[1] Bartal, Y. (1996, October). Probabilistic approximation of metric spaces and its algorithmic applications. In Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on (pp. 184-193). IEEE.

[2] Fakcharoenphol, J., Rao, S., and Talwar, K. (2003, June). A tight bound on approximating arbitrary metrics by tree metrics. In Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing (pp. 448-455). ACM.

[3] Heinonen, J. (2001). *Lectures on analysis on metric spaces.* Springer.

[4] Meyer, Y. (1992). *Wavelets and operators* (Vol. 37). Cambridge: Cambridge University Press.

[5] Rabinovich, Y., and Raz, R. (1998). Lower bounds on the distortion of embedding finite metric spaces in graphs. Discrete and Computational Geometry, 19(1), 79-94.

[6] Semmes, S. (1996). On the nonexistence of bilipschitz parameterizations and geometric problems about A_∞-weights. Revista Matemática Iberoamericana, 12(2).