5339 - Algorithm design under a geometric lens, Spring 2014, CSE, OSU
Homework 1
Instructor: Anastasios Sidiropoulos

Notation. Let $\mathbb{R}_{\geq 0}$ denote the set of non-negative real numbers.

Let $M = (X, \rho)$ be a metric space. That is, X is a set, and $\rho : X \times X \to \mathbb{R}_{\geq 0}$, satisfies the following conditions:

- For any $x, y \in X$, we have $\rho(x, y) = 0$ if and only if $x = y$.
- For any $x, y \in X$, we have $\rho(x, y) = \rho(y, x)$.
- For any $x, y, z \in X$, we have $\rho(x, y) \leq \rho(x, z) + \rho(z, y)$.

For any $r \geq 0$, a *r-partition* of M is a partition of X, such that every cluster has *diameter* at most r. That is, for any cluster C in the partition, and for any $x, y \in C$, we have $\rho(x, y) \leq r$.

Let $\beta > 0$. A (β, r)-*Lipschitz partition* of (X, ρ) is a distribution $\mathcal{D}$ over r-partitions of M, such that for any $x, y \in X$:

$$\Pr_{P \sim \mathcal{D}}[P(x) \neq P(y)] \leq \beta \cdot \frac{\rho(x, y)}{r}$$

Problem 1: Random embeddings without Steiner nodes. Let $M = (X, \rho)$ be a metric space, with $|X| = n$. Assume that for all $x, y \in X$, we have

$$1 \leq \rho(x, y) \leq \Delta,$$

for some $\Delta \geq 1$.

We have seen in class a proof that M admits a random embedding into a distribution over trees, with distortion $O(\log n \cdot \log \Delta)$. That is, there exists a distribution $\mathcal{D}$ over pairs (f_i, M_i), where every $M_i = (X_i, \rho_i)$ is the shortest path metric of some tree $T_i = (V_i, E_i)$, and $f_i : X \to X_i$, such that the following conditions are satisfied:

- The distances never decrease. That is, for any $x, y \in X$,

$$\mathbf{Pr}_{(f_i, M_i) \sim \mathcal{D}}[\rho_i(f_i(x), f_i(y)) \geq \rho(x, y)] = 1.$$

- In expectation, all distances do not increase by too much. That is, for any $x, y \in X$,

$$\mathbf{E}_{(f_i, M_i) \sim \mathcal{D}}[\rho_i(f_i(x), f_i(y))] \leq O(\log n) \cdot \rho(f(x), f(y)).$$

In the construction we discussed in class, for every tree T_i, there are vertices in V_i, that do not correspond to points in X_i. Such vertices are often referred to as *Steiner* in the literature. We will now see that Steiner vertices are not necessary.

Formally, you are asked to show that any metric (M, ρ) admits a random embedding as the above, but such that for any metric M_i, we have $X_i = V_i$. That is, the constructed trees do not have any Steiner vertices.

Recall that in the construction from class, any Steiner vertex of some tree T_i cannot be a leaf, since we always map the points of the metric *onto* (i.e. bijectively) the set of leaves of T_i. For a

node w in the tree, we define its *height* to be the minimum number of edges in a path from w to a descendant leaf node. Recall that every Steiner node of height $h > 0$ corresponds to some cluster of some randomly chosen 2^h-partition.

Suppose that we do the following: We assign an arbitrary *priority* to every point in M. I.e., we pick an arbitrary bijection $p : X \to \{1, \ldots, n\}$. Then, for every Steiner node v, corresponding to some cluster C, we simply replace the vertex C in V_i by the point $x \in C$ with maximum priority, i.e. such that $p(x)$ is maximized.

(a) Show that the resulting embedding is a valid random embedding into a distribution over trees.

(b) Show that the distortion remains $O(\log n \cdot \log \Delta)$.

Problem 2: From random trees to spanners. In this problem we will see that one can use a random embedding into a distribution over trees, to construct a sparse spanner.

Let $M = (X, \rho)$ be a metric space, with $|X| = n$. Assume that M, admits a random embedding into a distribution $\mathcal{D}$ over trees, with distortion $O(\log n)$, and without Steiner nodes, as in Problem 1.

Show that for any metric space $M = (X, \rho)$, there exists a graph $G = (X, E)$ (i.e. with vertex set X), with $|E| \leq O(n \cdot \log n)$, and such that for any $x, y \in X$, we have

$$\rho(x, y) \leq d_G(x, y) \leq O(\log n) \cdot \rho(x, y).$$

You may construct such a graph G as follows: Sample $c \cdot \log n$ pairs (f_i, M_i) from $\mathcal{D}$, independently, for some constant $c > 0$ to be determined later. Each such M_i is the shortest path metric of some tree $T_i = (X, E_i)$, i.e. with vertex set X. Add all the edges of all these tree to the graph G. That is, set

$$E = \bigcup_{i=1}^{c \cdot \log n} E_i.$$

Argue that for any pair of points $x, y \in X$, for any $i \in \{1, \ldots, c \cdot \log n\}$, when we sample T_i, the distance between x and y in T_i is at most $O(\log n)$ times the distance in M, with probability at least $1/2$. This follows by averaging over all choices for T_i.

Next, argue that if we sample k trees, then the distance between x and y is preserved up to a $O(\log n)$ factor in G, with probability at least $1 - 2^{-k}$.

Conclude that if we sample $k = c \cdot \log n$ trees, then the all distances are preserved in G up to a factor of $O(\log n)$, with positive probability, for sufficiently large constant $c > 0$. This implies in particular, that the desired graph G exists.

Problem 3: From random partitions to distance oracles. Let us consider the problem of quickly determining the distance between two given points in a metric space. Let $M = (X, \rho)$ be a metric space, with $|X| = n$, and assume that for any $x, y \in X$, we have

$$1 \leq \rho(x, y) \leq \Delta,$$

for some $\Delta \geq 1$.

We can easily determine the distance between two given points $x, y \in X$ in $O(1)$ time, simply by storing M as an $n \times n$ table of pairwise distances. However, this requires space $\Omega(n^2)$.

On the other hand, we can construct a *spanner* graph G as in Problem 2. Storing G requires space $O(n \cdot \log^{O(1)} n \cdot \log^{O(1)} \Delta)$. Using G, we can determine the distance between any two given points up to a multiplicative factor of $O(\log n)$. This however requires computing the shortest path in G between x and y, which can take time $\Omega(n \cdot \log^{O(1)} n)$.

A *distance oracle* is a data structure that simultaneously achieves small space, and fast query time. We will now see how to construct a distance oracle using random partitions.

For any $i \in \{0, \ldots, \log \Delta\}$, let $\mathcal{D}_i$ be a $(2^i, \beta)$-Lipschitz partition of M, for some $\beta = O(\log n)$. Let $k = c \cdot \log n$, for some constant $c > 0$ to be determined later. Suppose that you sample a collection of 2^i-partitions $P_{i,1}, \ldots, P_{i,k}$ from $\mathcal{D}_i$, independently.

For any $i \in \{0, \ldots, \Delta\}$, and for any $j \in \{1, \ldots, k\}$, you build a hash table $H_{i,j}$, where every point $x \in X$ is hashed to a "bucket" corresponding to $P_{i,j}(x)$, i.e. to the cluster of $P_{i,j}$ containing x. Argue that storing all these hash tables can be done in space $O(n \cdot \log^{O(1)} n \cdot \log^{O(1)} \Delta)$. Next, show that given two points $x, y \in X$, you can determine the distance $\rho(x, y)$, up to a multiplicative factor of $O(\log n)$, in time $O(\log^{O(1)} n \cdot \log^{O(1)} \Delta)$, and with high probability.

Problem 4: From random partitions to nearest neighbor search. Let $M = (X, \rho)$ be a metric space, with $|X| = n$, and assume that for any $x, y \in X$, we have

$$1 \leq \rho(x, y) \leq \Delta,$$

for some $\Delta \geq 1$.

Build a data structure that requires space $O(n \cdot \log^{O(1)} n \cdot \log^{O(1)} \Delta)$, and such that given a query point $x \in X$, it returns a point $y \in X$, such that

$$\rho(x, y) \leq O(\log n) \cdot \min_{z \in X : z \neq x} \{\rho(x, z)\},$$

with high probability, in time $O(\log^{O(1)} n \cdot \log^{O(1)} \Delta)$.