Sketching Algorithms for Big Data Fall 2017

Lecture 21 — November 14, 2017

Prof. Piotr Indyk *Scribe: Lucas Liebenwein*

1 Overview

In lecture 19, we saw streaming algorithms for geometric problems, such as minimum enclosing ball (MEB) and k-medians. We considered **insertion-only** streams and approached these types of problems using some kind of coreset, i.e., a weighted subset of the original data points that preserve the desired properties.

In this lecture, we will consider streaming algorithms for geometric problems for streams that allow both **insertions and deletions**. We will approach the problems using sketching algorithms, as opposed to coresets. The intuition behind this approach is that we can reduce the geometric problem to a linear sketching problem, i.e., we can represent the relevant quantities as vectors. Specifically, we will consider the following problems:

- Diameter of a set of points,
- Cost of minimum weight spanning tree (MST),
- Cost of minimum weight matching (MWM),
- Cost of minimum weight bi-chromatic matching (MWBM).

We will consider a set of points $P \in [\Delta]^2$, i.e., a **set of two-dimensional points with coordinates in the discrete set** $[\Delta]$. Note that we can easily extend the algorithms to work in any dimension d, however, we cannot lift the assumption about the coordinates coming from a discrete set. Moreover, the weight of an edge will denote the distance between points (e.g., ℓ_1-distance).

2 Diameter

Here we want to approximate the diameter D of a point set P, i.e., the maximum distance between any two points. More formally, we want to obtain a $(1 + \mathcal{O}(\varepsilon))$-approximation to D, where

$$D = \max_{p,p' \in P} D(p, p')$$

and $D(p, p')$ is the distance between two points $p, p' \in P$.

2.1 2-Approximation in Insertion-only Streams

We begin with presenting a trivial algorithm that takes $\mathcal{O}(1)$ space to compute a 2-approximation to D in an insertion-only stream. We initialize $\hat{D} \leftarrow 0$ and we maintain $\hat{D} \leftarrow \max\{\hat{D}, D(p_1, p_i)\}$ at

any time i, where p_i is the current element in the stream and p_1 is the first element in the stream. Then, we have that

$$\hat{D} \leq D \leq 2\hat{D}. \tag{1}$$

The lower bound in (1) follows from the definition of D, and the upper bound can be shown to hold using triangle inequality. Assume q, q' are the true points defining the diameter, then

$$D(q, q') \leq D(q, p_1) + D(p_1, q') \leq 2\hat{D}.$$

2.2 $(1 + \mathcal{O}(\varepsilon))$-approximation for insertion and deletion

Algorithm Consider the following algorithm for estimating the diameter:

Algorithm 1 Diameter approximation

$n_p^i(c) := |\{p \mid p \in c \wedge p \in P\}|, \forall c \in G_i, \forall i \in [m]$
function ApproximateD(P)
    $G_0, \ldots, G_m \leftarrow$ square grids with diameter $(1+\varepsilon)^{-log(1/\varepsilon)}, (1+\varepsilon)^1, (1+\varepsilon)^2, \ldots, 2\Delta$
    for $p \in P$ **do**
        Maintain linear sketch of $n_P^i,\ \forall i \in [m]$
    $i^* \leftarrow \min_{i \in [m]}\{i\}$ such that $\left\|n_P^i\right\|_0 \leq k = \mathcal{O}(\frac{1}{\varepsilon^2})$ ▷ $\left\|n_P^i\right\|_0$ from linear sketch
    Recover the set S of non-zero cells in $n_p^{i^*}$ ▷ using k-sparse recovery of a vector
    return $(1+\varepsilon)^{i^*} D(S)$ ▷ $D(S)$ is the diameter of the set S (grid coordinates)

Analysis We have already seen in previous lectures that we can maintain a linear sketch of n_P^i to obtain an estimate of the non-zero entries as well as how to recover n_P^i in case of k-sparsity. The remaining question is whether there exists a grid G_{i^*} such that there are at most k non-empty cells and how we choose it, otherwise the recovery of $n_P^{i^*}$ fails. We first show the existence:

Fact 1. *Let D be the diameter of a set of points P. Then there is always a grid level i such that*

$$(1+\varepsilon)^i \leq \varepsilon D \leq (1+\varepsilon)^{i+1}.$$

From Fact 1, it follows that D spans at most $k = \mathcal{O}(\frac{1}{\varepsilon^2})$ grid cells of the grid at level i. To obtain such i, we cannot use Fact 1 though, since it requires the knowledge of D. However, we can use our sketch for $\left\|n_P^i\right\|_0$ to estimate the sparseness of the grid at any level i. Since we want the best possible approximation we pick i^* to be the lowest grid level that has at most k non-empty cells. We further note that in any grid we make an error of at most $\mathcal{O}(\varepsilon)$ justifying the fact that we can approximate the diameter up to a factor of $(1 + \mathcal{O}(\varepsilon))$, see Fig. 1.

Space requirement Let n be the number of updates. The space requirement for this algorithm is $\mathcal{O}((1/\varepsilon^{\mathcal{O}(1)}) \operatorname{polylog}(n + \Delta))$ since each sketch requires $\mathcal{O}((1/\varepsilon^{\mathcal{O}(1)}) \operatorname{polylog}(n))$ space (as shown in Problem Set 1), and we require $\mathcal{O}(\operatorname{polylog}(\Delta))$ such estimators.

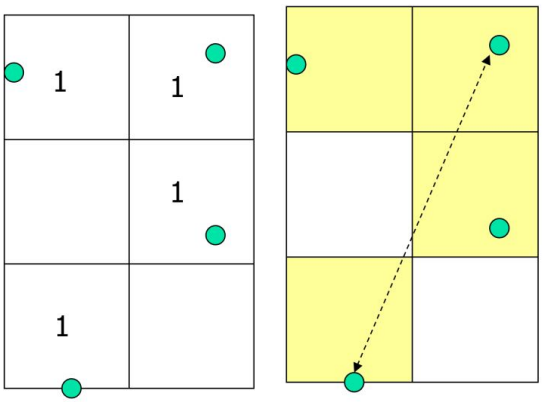

Figure 1: The diameter of a point set under its embedding.

3 Minimum Spanning Tree and Other Problems

We now turn our attention to a class of problems related to tree properties of a set of points, specifically we want to approximate the cost of the minimum spanning tree (MST), the cost of the minimum weight matching (MWM), and the cost of the minimum bi-chromatic weight matching (MBWM). These methods were first presented in [1].

3.1 Probabilistic Embedding

Before we discuss estimators for the aforementioned problems, we will introduce the quad-tree embedding. The intuition behind this data structure is that we maintain a tree-like data structure that approximately preserves distances between points. The idea of this embedding is due to [2]. Specifically, consider partitioning the space $[\Delta]^2$ into grid cells of diameter $2^0, 2^1, \ldots, 2\Delta$. We build the quad tree as follows:

1. Partition the initial cell of diameter 2Δ into 4 smaller cells,
2. Partition the resulting cells, which are non-empty, again into 4 smaller cells,
3. Recurse on steps 1. and 2. until all cells hold at most 1 item.

Note that since we assumed that all coordinates are from the discrete set $[\Delta]$, the tree height is bounded above by $\mathcal{O}(\log(\Delta))$. In Fig. 2, the procedure is shown for a set of points. Using random shifts of the underlying grids, we obtain the following useful properties of quad-trees:

Theorem 2. *Consider the quad-tree T of a set of points P constructed using grids that are shifted by a random vector $v \in [\Delta]^2$, then*

1. $\|p - q\|_1 \leq D_T(p, q)$, *and*
2. $\mathbb{E}[D_T(p, q)] \leq \|p - q\|_1 \mathcal{O}(\log(\Delta))$,

where $D_T(p, q)$ denotes the distance on the tree T between two points $p, q \in P$.

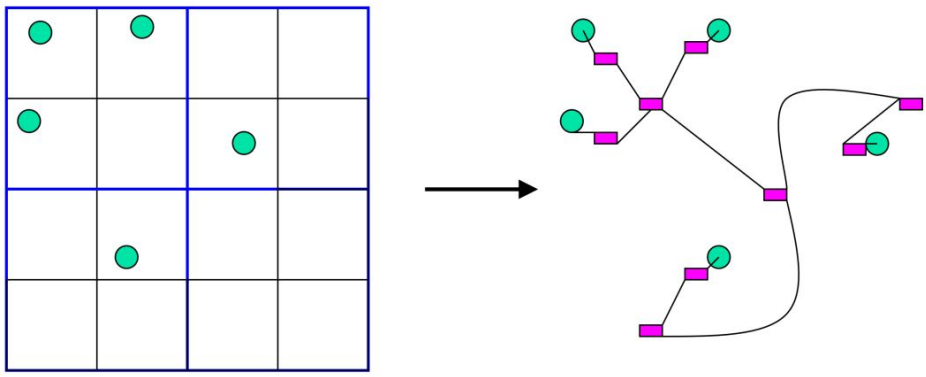

Figure 2: The quad-tree embedding for a set of points.

Proof. We will consider the two properties separately.

Property 1 This follows directly from the construction of the tree and does not even require random shifts. Let $c \in G_i$ be the smallest cell containing both p and q. Thus, $\|p-q\|_1 \leq 2^{i+1}$. Further, let $D_T(p,q)$ be the distance between the points p and q on the tree. Specifically, $D_T(p,q)$ is the sum of the weights of the edges that connect p and q. The weight of an edge is the grid diameter of its corresponding grid cell. Therefore,

$$D_T(p,q) = 2 \cdot 2^i = 2^{i+1} \geq \|p-q\|_1 .$$

Property 2 Consider some grid level i with cell side length 2^i and cells $c = (c_x, c_y)^T \in G_i$. Let $p = (p_x, p_y)^T \in P$ and $q = (q_x, q_y)^T \in P$. Then the probability that p and q belong to different cells $c, c' \in G_i$ along the direction of the x-axis is

$$\mathbb{P}(p_x \in c_x \wedge p'_x \in c'_x \neq c) = \min\left\{\frac{|p_x - q_x|}{2^i}, 1\right\} \leq \frac{|p_x - q_x|}{2^i}.$$

A similar results holds for the cells along the y-axis. Thus,

$$\mathbb{P}(p \in c \wedge q \in c' \neq c) \leq \frac{|p_x - q_x|}{2^i} + \frac{|p_y - q_y|}{2^i} = \frac{\|p-q\|_1}{2^i}.$$

Summing over all levels of the trees yields

$$\begin{aligned}\mathbb{E}[D_T(p,q)] &\leq \sum_{i=0}^{\mathcal{O}(\log(\Delta))} \underbrace{\frac{\|p-q\|_1}{2^i}}_{\mathbb{P}(p\in c\wedge q\in c'\neq c \,|\, c,c'\in G_i)} \underbrace{\mathcal{O}(2^i)}_{\text{contribution to } D_T(p,q) \text{ of layer } i} \\ &= \|p-q\|_1 \, \mathcal{O}(\log(\Delta)).\end{aligned}$$

□

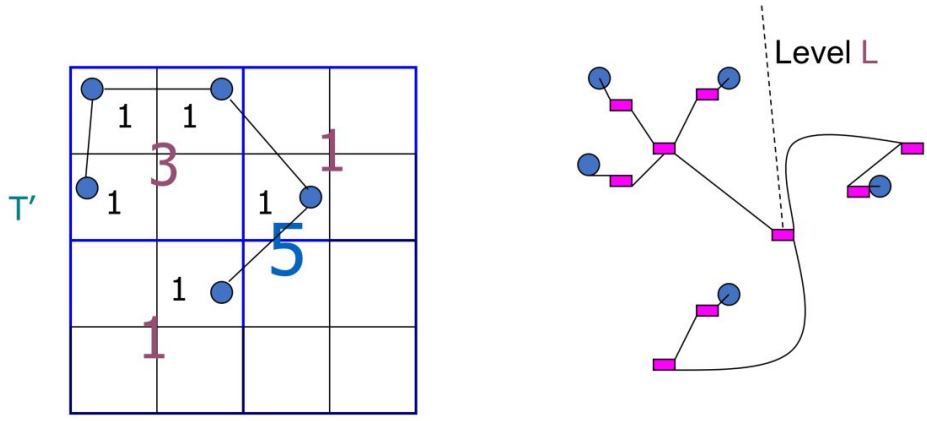

Figure 3: The minimum spanning tree and cost of its image under the quad-tree embedding.

3.2 Estimator for Minimum Spanning Tree

Let $n_P^i(c)$ be the count of points in cell c at grid level i. We maintain a linear sketch of $\left\|n_P^i\right\|_0$ for all levels i. Let L be be the lowest level with exactly one non-zero entry. Then

$$2\sum_{i=0}^{L-1} 2^i \sum_{c\in G_i} \left[n_p^i(c) > 0\right] = 2\sum_{i=0}^{L-1} 2^i \left\|n_P^i\right\|_0$$

is a $(1+\mathcal{O}(\log(\Delta)))$-approximation to the cost $cost(T')$ of the MST T' of a set of points P.

Proof. (Sketch) Let T denote the quad-tree of the point set P and T' the MST. Further, let T'' be the image of T' in T after removing duplicates, see Fig 3. Then, by the properties of the quad-tree embedding,

$$\mathbb{E}[cost(T'')] = \mathcal{O}(\log(\Delta))cost(T'). \tag{2}$$

Also, note that

$$cost(T') \leq 2cost(T''), \tag{3}$$

i.e., the cost of the MST is at most twice the cost of its image under the embedding. To derive this result, we can take a similar approach to the proof of Property 1 in Theorem 2. Let c_p and c_q be the largest cells containing p but not q, and q but not p, respectively, where p and q are two points connected in the MST. Then, $\|p-q\|_1 \leq 2D_T(c_p, c_q)$. $D_T(c_p, c_q)$ hereby denotes the distance between the two cells. Summing over all edges in the MST, we obtain (3). Combining (2) and (3) we have

$$cost(T') \leq 2\,\mathbb{E}[cost(T'')] \leq \mathcal{O}(\log(\Delta))cost(T').$$

Thus, $2cost(T'')$ is our desired estimator. But T'' is a subset of T and therefore

$$cost(T'') = cost(T \text{ up to level } L) = \sum_{i=0}^{L-1} 2^i \sum_{c\in G_i} \left[n_p^i(c) > 0\right].$$

□

3.3 Estimator for Minimum Weight Matching

A $(1 + \mathcal{O}(\log(\Delta)))$-approximation for the cost of the minimum weight matching is given by

$$\sum_i 2^i \sum_{c \in G_i} [n_P^i(c) \text{ is odd}].$$

This follows from the fact that the cost at some level is always twice as much as the cost one level below. Thus, we can take a greedy approach, where we match as many pairs as possible at each level. Odd leftovers are passed on to the next level and we then try to match as many pairs as possible at the next level. We repeat the procedure until we reach the highest level.

3.4 Estimator for Minimum Bi-chromatic Weight Matching

We shortly mention the $(1 + \mathcal{O}(\log(\Delta)))$-estimator for the miminum bi-chromatic weight matching, which is given by

$$\sum_i 2^i \sum_{c \in G_i} \left| n_G^i(c) - n_B^i(c) \right| = \sum_i 2^i \sum_{c \in G_i} \left\| n_G^i - n_B^i \right\|_1,$$

where $n_G^i(c)$ and $n_B^i(c)$ denote the count in cell $c \in G_i$ of green and blue points, respectively.

References

[1] Piotr Indyk. Algorithms for Dynamic Problems over Data Streams. *STOC*, 2004.

[2] Yair Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. *FOCS*, 1996.