Sketching Algorithms for Big Data Fall 2017

Lecture 19 – November 7, 2017

Prof. Piotr Indyk *Scribe: Kavya Ravichandran*

1 Overview

In the last lecture we saw an algorithm for graph sketching. In this lecture we consider streaming algorithms for geometric data of other types for insertion-only streams. The general approach to this type of problem is to construct a core-set, defined as a non-random sample of data that represents the whole data set.
Today, we considered the following problems and algorithms:

- develop a constant factor approximation algorithm requiring $O(\sqrt{nk})$ space for the **metric** k**-median problem** that utilized a "black-box" off-line algorithm.
- construct a core-set for the **minimum enclosing ball problem.**

2 Metric k-Median Problem

The metric k-median problem essentially asks us to cluster the points in the input and find k medians such that a defined cost function is minimized. We are given the following:

- "Oracle" access to a **metric function** $D(x, y)$ for points x, y in a metric space. This function satisfies standard properties of a metric function, including symmetry and the triangle inequality.
- Stream of metric points p defining a set S, with $|S| = n$.
- Objective defined as:

$$D(p, C) = \min_{c \in C} D(p, c)$$

$$\text{For } |C| = k, \text{cost}(S, C) = \sum_{p \in S} D(p, C)$$

$$\text{cost}(S, Q) = \min_{\substack{C \subseteq Q, \\ |C| = k}} \text{cost}(S, C)$$

Our goal, then is to approximate cost(S, S) and report the medians.

Intuition: We don't have much flexibility in what we store. We receive the points one after the other, and we are trying to solve the problem for general metric spaces. We are pretty much limited to storing a subset of points. All we have is the ability to calculate the distances between two points, but we really don't know anything else about the points. So let's do that!

2.1 Specification and Assumption

Specification We will develop a constant factor approximation algorithm that uses space $O(\sqrt{nk})$. Recursively applying this algorithm gives us an algorithm that takes $O(n^{\alpha}k)$. This approach is described in Guha, Mishra, Motwani, and O'Callaghan [1].

Assumption We will assume that there exists an *offline* $b-$approximate algorithm that uses linear space and works for the weighted version of the problem. (Note that we can show that it is NP-hard to provide an exact solution to this problem.) Indeed, Arya et al.[2] show a $(3+\varepsilon)-$approximation algorithm for this problem, so this is not a futile assumption.

2.2 Algorithm

Intuition "Medians of weighted medians are approximate medians."

Statement Our algorithm makes use of the $b-$approximate algorithm that requires linear space. We begin by considering the stream in blocks $S_1, ..., S_L$, where $L = \sqrt{\frac{n}{k}}$. This means $|S_i| = \sqrt{nk}$. For each S_i, we first find medians $c_1^i, ..., c_k^i$ which $b-$approximate $\text{cost}(S_i, S_i)$. Then, we compute m_j^i representing the number of points in S_i that have been assigned to c_j^i (we will refer to this as "Phase 1"). Now, we find the $b-$approximate k medians C' for the weighted set $MC = \{m_1^1 c_1^1, ..., m_k^L c_k^L\}$ (we will refer to this as "Phase 2").

2.3 Proof

Intuition Apply triangle inequality many times!

Notation $C =$ the optimum set of medians, such that $\text{cost}(S, C) = \text{cost}(S, S)$.

We proceed by first bounding $\text{cost}(S, S)$ and bounding the outcome of Phase 1 by this. Then, we consider

---

Claim 1 For any Q, not necessarily a subset of S, $\text{cost}(S, S) \leq 2\text{cost}(S, Q)$.

Proof We can replace each median by the closest point in S. Then by the triangle inequality, as we can see in Figure 2.3, each point is at most twice as far away from the orange point as from the red point. Thus, $\text{cost}(S, S) \leq 2\text{cost}(S, Q)$

Claim 2 $\sum_i \text{cost}(S_i, S_i) \leq 2 \cdot \text{cost}(S, S)$

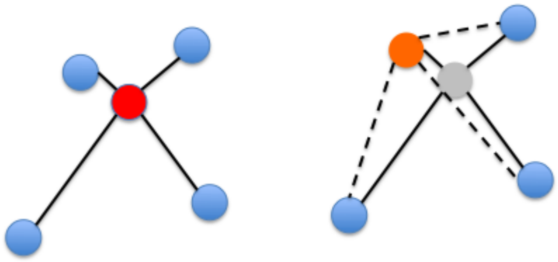

Figure 1: The red point in the left half is the local median chosen by the algorithm. The closest point to it is the one denoted in orange in the right half of the diagram. By triangle inequality, each point is no more than twice as far away from the orange point as from the red point.

Proof From Claim 1, we have $\text{cost}(S_i, S_i) \leq 2 \cdot \text{cost}(S_i, Q)$. Therefore,

$$\sum_i \text{cost}(S_i, S_i) \leq 2 \cdot \sum_i \text{cost}(S_i, S_i) \leq 2 \cdot \sum_i \text{cost}(S_i, C) = 2 \cdot \text{cost}(S, S)$$

.

Corollary The algorithm will find $(nk)^{\frac{1}{2}}$ medians MC with cost at most $2b$ cost(S, S).

With that, we have bounded the cost in Phase 1. Now, we consider Phase 2.

Claim 3 $\text{cost}(MC, MC) \leq 2(2b \text{ cost}(S, S) + \text{cost})$

Proof We start by bounding cost(MC, C), where C is the optimal set of medians.

Notation

- $q \in MC$ a single point, possibly out of many duplicates
- $p \in S$ assigned to q in the algorithm's solution to MC
- $c \in C$ optimal median to which p is assigned in the optimal solution.

We once again apply the triangle inequality; we can connect each q to c through p. Our total cost breakdown is as follows:

- $\forall q$ to p, the cost is $2b$ cost(S, S)
- $\forall p$ to c, the cost is cost(S, S)

Therefore, $\text{cost}(MC, C) \leq 2b\ \text{cost}(S, S) + \text{cost}(S, S)$, so $\text{cost}(MC, MC) \leq 2\ \text{cost}(MC, C) \leq 2(2b\ \text{cost}(S, S) + \text{cost}(S, S))$.

Altogether, in Phase 1, we connect the stream S to the set of intermediate medians MC, and the cost is upper bounded by $2b\ \text{cost}(S, S)$, and in Phase 2, we connect MC to the set of calculated k medians, an the cost here is upper bounded by $b \cdot 2(2b\ \text{cost}(S, S) + \text{cost}(S, S))$. The total cost is then $\leq \boxed{4b(b+1)\ \text{cost}(S, S)}$.

2.4 Comments

We can execute Phase 1 of this algorithm in a distributed fashion.

3 Core-Sets

3.1 Setup

We are given a set of points P, and our goal is to **minimize a function $C_p(o)$, where o is a solution**.

In this lecture, we consider the minimum enclosing ball problem. Our objective function is defined $C_p(o) =$ the smallest radius of a ball centered in o containing P.

Definition $S \subseteq P$ is a (weak) $c-$core-set for P if, for any o in the space:

$$C_s(o) \leq C_p(o) \leq c \cdot C_s(o) \tag{1}$$

Assuming monotonicity, $C_A(o) \leq C_B(o)$ if $A \subseteq B$. This means that the first inequality in 1 is trivially true.

3.2 Core-Set for Minimum Enclosing Ball (MEB)

Intuition Remembering multiple points along the same vector is redundant, since we only need to know how far the farthest one is. We compute extremal points in "all" directions.

We construct a core-set for MEB as follows:

- choose "densely" spaced directions $v_1, ..., v_k$; i.e., for any u, there is a v_i such that

$$\text{angle}(u, v_i) \leq \alpha$$

- for each direction, maintain the extremal point.

In $\mathbb{R}^d, k = O(\frac{1}{\alpha})^{d-1}$ directions suffice. We claim that the resulting set is a $1 + O(\alpha^2)-$core-set for MEB. This then gives us a $1+O(\varepsilon)$-approximate streaming algorithm for MEB storing $O(1/(\varepsilon^{\frac{d-1}{2}}))$ points.

3.3 Proof

Consider the smallest ball B containing S centered at o. Assume the radius is 1. We must show that:

$$C_p(o) \leq (1 + O(\alpha^2))C_s(o)$$

i.e., that the points in $P - S$ (say q) cannot be too far from B. We have

$$\cos\left(\frac{\alpha}{2}\right) \leq \frac{1}{1+\varepsilon} \approx 1 - \varepsilon.$$

Using a Taylor expansion, we know that $\cos(\alpha) \approx 1-\alpha^2$, so $\varepsilon = O(\alpha^2)$. Thus, we get a $1+O(\varepsilon)$-core set for MEB of size $O(1/(\varepsilon^{\frac{d-1}{2}}))$.

References

[1] Sudipto Guha, Nina Mishra, Rajeev Motwani, Liadan O'Callaghan. Clustering Data Streams. *Proceedings 41st Annual Symposium on Foundations of Computer Science*, 359–366,2000.

[2] Local search heuristic for k-median and facility location problems. *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, 21-29,2001.