CSE 522: Sublinear (and Streaming) Algorithms **Spring 2014**

Lecture 14: Streaming using Coresets. Clustering

May 14, 2014

Lecturer: Paul Beame *Scribe: Paul Beame*

1 Streaming using Coresets

Recall the Reduce Property satisfied by all coresets: If S is an α-coreset for P and S' is a β-coreset for S then S' is an $\alpha\beta$-coreset for P.

Also recall the disjoint union property satisfied by some cost functions C such as the Minimum Enclosing Ball (MEB) cost function: IF S is an α-coreset for P, S' is an α-coreset for Q, and $P \cap Q = \emptyset$ then $S \cup S'$ is an α-coreset for $P \cup Q$.

Theorem 1.1. *Suppose that for every $\varepsilon > 0$, cost function C has coresets of size at most $A(\varepsilon)$ and has the disjoint union property, then for every data stream σ of points, there is a data stream algorithm using space $O(A(\varepsilon/\log n)\ \log n)$ that returns a $(1+\varepsilon)$-approximation to $\inf_{x\in\mathbb{R}^d} C_\sigma(x)$.*

Proof. The algorithm will compute the elements of a tree of recursive coresets that provide successively smaller approximations for the point set given by σ. Let $\gamma = \varepsilon/\log_2 n$ and fix a "buffer size" parameter B. If $\sigma = y_1 \dots y_n$. The algorithm will be based on a tree over these n elements that has fan-in B at the bottom level and binary fan-in at all higher levels. It will have height $\log_2(n/B)+1$. Each internal node of the tree will compute a $(1+\gamma)$-coreset for the union of the values of its children and hence each such node corresponds to a set of size at most $A(\gamma) = A(\varepsilon/\log n)$. We can think of the sequence of coresets at each level above the leaves as a separate stream. Whenever the coreset associated with a node has been computed, all the elements in its subtree are deleted from the memory. At the end of the algorithm, the only item remaining in storage will be the coreset associated with the root. In particular this means that at any time, the memory contains at most B elements from the input, and at most $2A(\gamma)$ elements from each other level of the tree. Therefore the total storage is at most $O(B + A(\varepsilon/\log n)\ \log(n/B))$ which is $O(A(\varepsilon/\log n)\log n)$ for $B = A(\varepsilon/\log n)$.

With this value of B, the $(1+\gamma)$-coreset for each node one level above the leaves can simply be the set consisting of its children. (It is 1-coreset for its children but we won't use that fact.)

We prove inductively that any set h-levels above the leaves is at $(1+\gamma)^h$-coreset for the set of inputs labelling the inputs in its subtree. As above, this is true for $h = 1$.

Given two $(1+\gamma)^h$-coresets S_L and S_R for sets P_L and P_R, by the disjoint union property, $S_L \cup S_R$ is an $(1+\gamma)^h$-coreset for $P_L \cup P_R$ and by the recursive property of coresets, a $(1+\gamma)$-coreset S for $S_L \cup S_R$ is a $(1+\gamma)^{h+1}$-coreset for $P_L \cup P_R$.

Since

$$\begin{aligned}(1+\gamma)^{\log_2(n/B)+1} &= (1+\varepsilon/\log_2 n)^{\log_2(n/B)+1} \\ &\leq e^{(\varepsilon/\log_2 n)(\log_2 n - \log_2 B + 1)} \\ &\leq (e^{\varepsilon})^{1-(\log_2 B+1)/\log_2 n} \\ &\leq 1+\varepsilon\end{aligned}$$

the final set S at the root that remains at the end is a $(1+\varepsilon)$-coreset for σ. The algorithm then computes $\inf_{x\in\mathbb{R}^d} C_S(x)$ which is a $(1+\varepsilon)$-approximation to $\inf_{x\in\mathbb{R}^d} C_\sigma(x)$. □

In particular this implies

Corollary 1.2. *There is a data stream algorithm that produces a $(1+\varepsilon)$-approximation for the Minimum Enclosing Ball problem in d dimensions that uses space $O(\varepsilon^{-(d-1)/2}\log^{(d+1)/2} n)$.*

In particular for $d = 2$ this requires space $O(\varepsilon^{-1/2}\log^{3/2} n)$. On the problem set you will show that in the special case of our coreset construction the $\log^{(d+1)/2} n$ factor in the space bound can be removed.

2 Clustering

In some sense the MEB problem can be seen as approximating a point set by a single center together with its enclosing ball. We now consider algorithms that approximate a point set based on k different elements, which we can think of as centers of clusters.

Definition 2.1. *A* metric space *is pair (M,d) where $d : M \times M \to \mathbb{R}_+$ satisfies*

1. *$d(x,y) = 0$ iff $x = y$. (If $d(x,x) = 0$ but it is possible to have $d(x,y) = 0$ even if $x \neq y$ then this is called a* semi-metric. *Most algorithms work equally well for semi-metrics.*
2. *(Symmetry) $d(x,y) = d(y,x)$ for all $x, y \in M$.*
3. *(Triangle Inequality) $d(x,y) \leq d(x,z) + d(z,y)$ for all $x, y, z \in M$.*

Examples:
1. $d(x, y) = ||x - y||_p$ for some $p \geq 0$.
2. $d(x, y) =$ length of shortest path between vertices x and y in undirected graph G with non-negative edge weights.

For problems on metric spaces, we will assume that the stream σ consists of a sequence of elements of M. We measure space in terms of the number of data points in M we store as well as the number of bits of working storage.

We wish to summarize a data set using a small number of points. For a given set $S \subseteq M$ define

$$d(x, S) = \min_{y \in S} d(x, y).$$

We consider several different measures of the quality (or cost) of a set $R \subseteq M$ of $\leq k$ representative points for a stream σ by

- $\Delta_\infty(\sigma, R) = \max_{x \in \sigma} d(x, R)$, which is k-*center* objective.
- $\Delta_1(\sigma, R) = \sum_{x \in \sigma} d(x, R)$, which is the k-*median* objective and has been studied especially when $d(x, R) = ||x - y||_1$.
- $\Delta_2(\sigma, R) = \sum_{x \in \sigma} d(x, R)^2$, which is the k-*means* objective and has been studied especially when $d(x, R) = ||x - y||_2$.

With all of the above, the set R of size k that minimizes the objectives may include points of M that are not in σ.

THere are a number of streaming algorithms that produce good k-median and k-means $(1 + \varepsilon)$-approximations using small space, particularly using coresets (sometimes in a more general version that allows the small weighted set to include points outside of σ). These algorithms typically involve some computation of weight values for the points in σ and sampling the elements of the stream with probabilities proportional to those weights. These algorithms often have space requirements whose dependence on ε and d is of the form $O(1/\varepsilon^d)$. However, more recent algorithms can achieve space bounds whose dependence on d is polynomial but whose dependence on ε is worse, of the form $2^{k/\varepsilon}$. These algorithms would take up too much time to present. We focus on a clean simple algorithm for k-center approximation.

k-Center Approximation The k-center problem is NP-hard and even approximating the optimum k-center better than a factor 2 is NP-hard, so we will not be able to achieve that. We will give a deterministic algorithm due to Charikar, Chekuri, Feder, and Motwani that produces an 8-approximation. Guha has developed a more complicated algorithm that achieves $(2 + \varepsilon)$-approximations. The basic ideas are somewhat similar to the thresholds that we will use in the algorithm we discuss but it uses a similar cascaded structure as the generic coresets-based algorithm.

Lemma 2.2. *If $x_1, \ldots, x_{k+1} \in \sigma \subseteq M$ satisfy $d(x_i, x_j) \geq t$ for all $i \neq j$ then for all $R^* \subseteq M$ with $|R| \leq k$,*

$$\Delta_\infty(\sigma, R^*) \geq t/2.$$

Proof. Consider the mapping that sends each of $x_1, \ldots, x_{k+1}$ to its closest point in R^*. By the pigeonhole principle there exist $x_i \neq x_j$ and $r \in R^*$ such that $d(x_i, R^*) = d(x_i, r)$ and $d(x_j, R^*) = d(x_j, r)$. Therefore

$$2d(\sigma, R^*) \geq d(x_i, R^*) + d(x_j, R^*) = d(x_i, r) + d(x_j, r) \geq d(x_i, x_j) \geq t,$$

which is what we needed to show. □

Doubling Algorithm for k-Center

1: Initialize:
2: $S \leftarrow$ First $k+1$ points in the data stream
3: $(x, y) \leftarrow$Closest pair of points in S
4: $\tau \leftarrow d(x, y)$
5: $R \leftarrow S \setminus \{x\}$
6: Process:
7: **for** each $i > k+1$ **do**
8: **if** $\min_{r \in R} d(x_i, r) \geq 2\tau$ **then**
9: $R \leftarrow R \cup \{x_i\}$
10: **while** $|R| > k$ **do**
11: $R \leftarrow$Maximal subset $R' \subseteq R$ such that $d(r, s) \geq 2\tau$ for all $r \neq s \in R'$
12: $\tau \leftarrow 2\tau$
13: **end while**
14: **end if**
15: **end for**
16: Output: R

Lemma 2.3. *The following invariants for the Doubling Algorithm hold each time step 7 is executed and when the algorithm terminates*

(1) $\forall r \neq s \in R$, $d(r, s) \geq \tau$,

(2) $\Delta_\infty(\sigma, R) \leq 2\tau$,

and the following invariant holds immediately after step 5 and immediately prior to each execution of step 11,

(3) $\forall R^* \subseteq M$ *with* $|R^*| \leq k$, $\Delta_\infty(\sigma, R^*) \geq \tau/2$.

Proof. We prove these by induction. In the base case, (1) is immediate from the fact that τ is the minimum distance in $S \supset R$. (2) follows from the fact that at this point $\sigma = S$ and so $\Delta_\infty(\sigma, R) = d(x, R) = \tau \leq 2\tau$. (3) follows from the previous lemma applied to the set S.

For the induction step we have two cases depending on the value of $\min_{r \in R} d(x_i, r)$.

If $\min_{r \in R} d(x_i, r) \leq 2\tau$, then the process loop makes no changes to R or τ and

$$\Delta_\infty(\sigma \cup \{x_i\}, R) = \max\{\Delta(\sigma, R), d(x_i, R)\} \leq 2\tau$$

using the inductive hypothesis (2) at the start of the iteration.

If $\min_{r \in R} d(x_i, r) > 2\tau$, then $\Delta_\infty(\sigma \cup \{x_i\}, R \cup \{x_i\} = \Delta_\infty(\sigma, R) \leq 2\tau$ so step 9 preserves property (2). Further, since $\min_{r \in R} d(x_i, r) > 2\tau$ and the inductive hypothesis for (1) together imply that step 9 preserves property (1). The only issue is that R might be too large Indeed we must have $|R| = k + 1$ just before step 11 is reached. Then the previous lemma, together with (1) imply (3) holds .

It remains to show that steps 11 and 12 together preserve properties (1) and (2). The fact that they preserve (1) is immediate from fact that all points in R' in line 11 are of distance at least 2τ apart which is still at least τ after the value of τ is doubled in step 12.

To see that they preserve (2), let $x \in \sigma$ be arbitrary. Then by definition $d(x, R) \leq \Delta_\infty(\sigma, R) \leq 2\tau$ by (2) just before step 11 is executed.

Let r be the closest point in R to x.

If $r \in R'$ then $d(x, R') \leq d(x, r) \leq 2\tau$

If $r \notin R'$ then because $R' \subseteq R$ is maximal, there must be some point $s \in R'$ such that $d(r, s) < 2\tau$. Now $d(x, R') \leq d(x, s)$ by definition since $s \in R'$. Since $s \in R'$, we have

$$\begin{aligned} d(x, R') &\leq d(x, s) \\ &\leq d(x, r) + d(r, s) \qquad \text{by the triangle inequality} \\ &< 2\tau + 2\tau = 4\tau \end{aligned}$$

Therefore after step 11 we have $\Delta_\infty(\sigma, R) < 4\tau$ and by property (3), which holds just before step 11, $\Delta_\infty(\sigma, R^*) \geq \tau/2$ implies that $\Delta_\infty(\sigma, R) < 8\Delta_\infty(\sigma, R)$ and this relationship is unaffected by step 12. The doubling of τ in step 12 has the effect of the changing the bound $\Delta_\infty(\sigma, R) < 4\tau$ to $\Delta_\infty(\sigma, R) < 2\tau$ which implies that (2) holds after each iteration of the process loop. It is also immediate that $|R| \leq k$ after each iteration of the process loop. □

This immediately proves the following theorem.

Theorem 2.4. *The Doubling Algorithm uses space $O(k)$ and produces a factor 8 approximation to the optimum k-center for any k.*

Note that though the true optimum may require choosing an element of the metric space not present in the input stream, the above calculations show that for approximation purposes, it suffices to choose $R \subseteq \sigma$. =