Better Streaming Algorithms for Clustering Problems

Sachin Goel

May 1, 2015

1 Overview

In this paper, we discuss a polylog space algorithm to solve k-median problem in a streaming setting. The algorithm proceeds in phases, consuming at least one point in each phase, bounding the number of phases by $\#X$. This algorithm uses an *Online facility location* algorithm as a sub-routine which is discussed in section 2.

2 Online-FL Algorithm

An instance of the *Facility Location* problem is a set of *demand* points and a set of *feasible facilities* with each feasible facility associated with a *facility cost* and a distance function between all pairs of points. The objective is to pick a subset of feasible facilities such that the sum of costs of the facilities chosen plus the sum of distances of demand points from the nearest facilities is minimized. In other words, if $\mathcal{D}$ is the set of demand points and $\mathcal{F}$ is the set of feasible facilities with a facility cost of c_f associated with each facility c in $\mathcal{F}$, then:

$$min_{F\subseteq\mathcal{F}} \sum\nolimits_{f\in F} c_f + \sum\nolimits_{x\in\mathcal{D}} d(x,F)$$

where $d(x,F)$ is the distance of x from the nearest point in F.
The above problem is a Lagrangian relaxation to the k-median problem in which the number of clusters(medians) is not fixed, rather there is an additive cost associated with each cluster. We now discuss an algorithm for solving the online version of the facility location problem, with uniform costs, which is due to Meyerson.

Algorithm Online-FL(data stream X, facility cost f).

1. Make one pass over X performing the following steps for each $x \in X$.
2. Let δ be the distance of the current point x to the closest *already-open* facility.
3. With probability δ/f (or probability 1 if $\delta/f > 1$), open a new facility at x otherwise assign x to the closest already-open facility.

Consider an instance of the k-median problem over a point stream X and let OPT be its optimal solution. Let L be a lower bound on this optimal solution and define a facility cost $f = \frac{L}{k(1+\log n)}$ [The intuition is that we are going to distribute the cost equally among an expected $k \log n$ medians.]

Lemma 1.

The number of medians Online-FL is expected to produce is at most $k(1+\log n)(1+\frac{4OPT}{L})$. The expected cost is at most $L + 4OPT$

Proof. Let $c_1^*, c_2,^* \ldots, c_k^*$ be the medians in the optimal solution. Let d_p^* denote the distance of point p from the nearest median in the optimal. Let C_i^* be the cluster associated with the median c_i^* and A_i^* be the cost associated with the cluster C_i^*, i.e. $A_i^* = \sum_{x \in C_i^*} d(x, c_i^*)$. Define $a_i^* = \frac{A_i^*}{|C_i^*|}$.
Consider an optimal cluster C_i^*. Define set $S_j = \{p | p \in C_i^*, 2^{j-1} a_i^* < d_p^* < 2^j a_i^*\}$ for $j = 1, 2, \ldots, \log n$. Note that for $j > \log n$, S_j contains points p st. $d_p^* > n a_i^*$ which is not possible. For the points in S_j, the expected cost before a facility is opened at some point in S_j is at most f. However, as soon as a facility is opened at a point in S_j, the maximum distance of a point $p \in S_j$ from an already-open facility is at most $d_p^* + 2^j a_i^* = d_p^* + 2.2^{j-1} a_i^* < 3d_p^*$.
Now, for the points in $P = C_i^* \setminus \cup_{j=1,2,\ldots,\log n} S_j = \{p | d_p^* < a_i^*\}$, the expected cost before a facility is opened at some point in P is at most f. However, as soon as a facility is opened at a point in P, the maximum distance of a point $p \in P$ from an already-open facility is at most $a_i^* + d_p^*$.
Thus, the expected number of medians opened amongst points in C_i^* is at most $1 + \log n + \frac{1}{f} \sum_{p \in C_i^*} (a_i^* + 3d_p^*) \leq 1 + \log n + \frac{4A_i^*}{f}$. Summing this over all clusters $C_i^*, i = 1, 2, \ldots, k$, expected number of medians is at most $k(1+\log n)(1+\frac{4OPT}{L})$.
Expected cost for points in C_i^* is at most $f(1+\log n) + \sum_{p \in C_i^*} 3(a_i^* + d_p^*) \leq \frac{L}{k} + 4A_i^*$. Summing this over all clusters $C_i^*, i = 1, 2, \ldots, k$, expected number of medians is at most $L + 4OPT$. □

Corollary 1.

With probability at least 1/2, algorithm Online-FL produces a solution of cost at most $4(L + 4OPT)$ and using at most $4k(1+\log n)(1+\frac{4OPT}{L})$ medians.

3 Streaming k-median

We now present the details of the algorithm for finding medians in polylog space.
As mentioned earlier, the algorithm proceeds in phases. Each phase consumes some points from the data stream X. During phase i, the algorithm takes a modified input X_i which is the remaining data stream and the medians found in the previous phase, along with their associated weights(which are defined as the number of points associated with them). It then outputs medians along with their associated weights and this continues till the whole of data stream is consumed.

During each phase, the algorithm processes some points and marks them as *read*. We'll make a distinction later that the point which leads to a phase change is not marked *read* rather only *seen*.

Algorithm Polylog-space(point stream X, integers k and n).

1. $L_1 \leftarrow$SET-LB$(X)/\beta$
2. $i \leftarrow 1; X_1 \leftarrow X$
3. While there are unread points in X:
   (a) $M_i \leftarrow$PARA-CLUSTER(L_i, X_i, k, n)
   (b) $X_{i+1} \leftarrow M_i || (X - R_i)$
   (c) $L_{i+1} \leftarrow \beta L_i$
   (d) $i \leftarrow i + 1$
4. Return M_{i-1}, the medians given by the most recent invocation of PARA-CLUSTER

Algorithm SET-LB(point stream X, integer k).

return the distance between the closest pair of points between the first $k+1$ members of X

Algorithm PARA-CLUSTER(point stream S,lower bound L,integers k and n).

1. Run $2 \log n$ parallel invocations of Online-FL with a facility cost of $f = L/(k(1+\log n))$ on the input stream S
2. Each invocation is run as long as the number of medians doesn't exceed $4k(1+\log n)(1+4(\gamma+\beta))$ and the cost of solution, on the point stream S doesn't exceed $4L(1+4(\gamma+\beta))$ where γ and β satisfy the inequality $\gamma + 4(1 + 4(\gamma + \beta)) \leq \gamma\beta$. If either condition is violated, that invocation is stopped.
3. When all the invocations have stopped, mark the points processed(seen) by the last-finishing invocation as read **except** for the last point which resulted in the violation of the conditions. Return the medians found by this invocation.

Note that:

1. The number of points *seen* is always one more than the number of points *read* except in the last phase when the algorithm terminates because there are no more points left.
2. The last-finishing invocation is the one which *saw* the most number of points from the point stream S
3. The medians returned by the PARA-CLUSTER algorithm are weighted points, and the Online-FL adapts to this by using a probability $w\delta/f$ to open a facility at a point. Note that the point in the original point stream all have weights 1.

4. γ and β are kept fixed for the full run of the algorithm. They must satisfy the inequality $\gamma + 4(1 + 4(\gamma + \beta)) \leq \gamma\beta$.

Definitions:

1. R_i is defined as the set of points *read* during phases $1, 2, \ldots, i$ and we define S_i as the set of points *seen* during the phases $1, 2, \ldots, i$.

2. Define A_i as the cost of the medians M_{i-1} on the set R_{i-1}, $A_1 = 0$.

3. Let p be the number of phases. For $1 \leq i < p$, define OPT_i as the cost of optimal solution for S_i when the medians are allowed to be chosen from X.

The idea is to maintain a lower bound on the optimal cost of the points seen so far, which is exactly what L_i maintains.

Lemma 2.

Let $P(i)$ denote the event that $A_i \leq \gamma L_i$, and let $Q(i)$ denote the event that $L_{i+1} = \beta L_i \leq OPT_i$. For appropriate values of γ and β, with probability $1 - \frac{1}{n^2}$, for all $1 \leq i < p$, $P(i)$ and $Q(i)$ hold.
Let us now see how we can obtain a constant factor approximation to the k-median problem on X using this.
Note that $Q(p-1)$ and $P(p-1)$ hold with probability $1 - \frac{1}{n}$. $L_{p-1} \leq OPT_{p-1}/\beta \leq OPT_p/\beta$, which implies $L_p = \beta L_{p-1} \leq OPT_p$. Now, the last phase of the algorithm outputs $O(k \log n)$ medians; let's call this set S. The cost of these medians is at most $4L_p(1 + 4(\gamma + \beta)) \leq 4(1 + 4(\gamma + \beta))OPT_p = \alpha OPT$.
It is easy to see that there must exist a solution on S with k medians amongst points in S with cost at most $2(1 + \alpha)OPT$. Using a c-approximation algorithm to solve the k-median problem on S, we obtain a solution of cost $2c(1 + \alpha)OPT$ and the cost of this solution on the point stream X is at most $\alpha OPT + 2c(1 + \alpha)OPT = (\alpha + 2c(1 + \alpha))OPT$. We thus obtain a constant factor approximation for the k median problem.

Note that in the above algorithm there is no guarantee that the number of phases is at most n. To ensure this, we need to obtain a better bound on L_i. Suppose we ensure that $OPT_i \leq \beta L_{i+1}$. This means that at the termination of phase $i + 1$, we will have definitely read at least one point from X with high probability. To do this, we compute an approximate k-median solution on M_i and x where x is the point which was marked *seen*, not *read*. Suppose we use a c-approximation algorithm and let APX_i be the value of the c-approximate solution thus obtained. Then, we set L_{i+1} as follows:

$$L'_{i+1} = \frac{APX_i}{2c(1 + \gamma)}$$

$$L_{i+1} = max(\beta L_i, L'_{i+1})$$

with an added constraint $\beta \geq 2c(1 + \gamma) + \gamma$