CS 514: Advanced Algorithms II – Sublinear Algorithms **Rutgers: Spring 2020**

Lecture 10

April 03, 2020

Instructor: Sepehr Assadi *Scribe: Chen Wang*

1 Clustering: An Introduction

Consider the following problem: you are given some data points with *no* additional knowledge on them other than *observations* (e.g. values, distributions, manifold). In this case, how can you characterize the similarities and differences between each of them? A natural way one can adopt is to 'group' them. For example, if you are given a bunch of candy bars and potato chips, you will usually put them into their respective boxes without external instructions. This kind of decision is based purely on observations (e.g. appearance, flavor), and the same idea leads us to the clustering algorithms.

In clustering algorithm, we are given a bunch of points $p_1, p_2, \cdots, p_n \in \mathbb{R}^d$, and we want to group 'similar' points to the same cluster. Here, the 'similarity' is characterized by the distance between points with certain metrics. Furthermore, in the sense of 'grouping', we aim to find K centers $C_1, C_2, \cdots, C_K$ such that if we assign each point to its closest center, the distance metric will be minimize. Formally, the above process can be given as the following:

- Given a set of points $P = \{p_1, \cdots, p_n\} \subseteq \mathbb{R}^d$ and unknown centers $C_1, C_2, \cdots, C_K \in \mathbb{R}^d$.
- For each point p_i, define the nearest center as $C_{(p_i)} = \arg\min_{C \in \{C_1, C_2, \cdots, C_K\}} \|p - C\|$.
- Minimize the cumulative distance with certain metric $\sum_{i=1}^{n} \|p_i - C_{(p_i)}\|$.

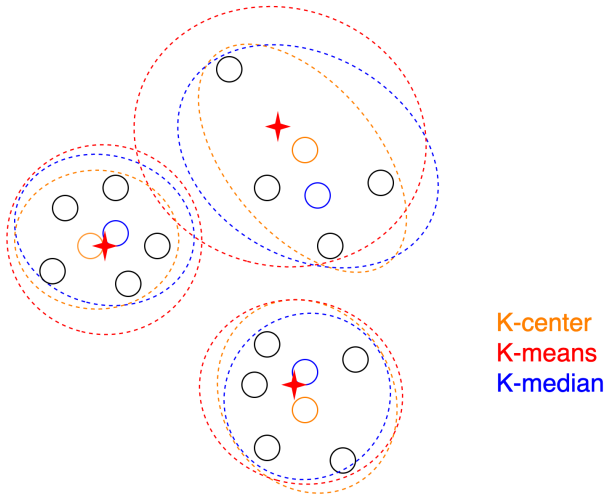

Figure 1: Clustering with $K = 3$ for K-center, K-means, and K-median clustering objectives.

An illustration of clustering can be shown as figure 1. Depending on the metric $\|\cdot\|$ we are using, there are different types of clustering. The commonly-used clustering methods include K-center, K-median, and K-means. Their respective optimization objective can be given as the following:

- K-Center. The optimization target is the maximum distance between the picked point and the points in the cluster:
$$C_1, \cdots, C_K = \underset{\{C_1,\cdots,C_K\}}{\arg\min} \max_{p\in P} \|p - C_{(p)}\|_2$$
- K-Median. The optimization target is the median distance between the picked point and the points in the cluster:
$$C_1, \cdots, C_K = \underset{\{C_1,\cdots,C_K\}}{\arg\min} \sum_{p\in P} \|p - C_{(p)}\|_2$$
- K-Means. The optimization target is the square of the Euclidean distance between the clustering center and the points in the cluster:
$$C_1, \cdots, C_K = \underset{\{C_1,\cdots,C_K\}}{\arg\min} \sum_{p\in P} \|p - \bar{C}_{(p)}\|_2^2$$
Notice that the notation $\bar{C}_{(\cdot)}$ instead of $C_{(\cdot)}$ denotes that the clustering center is *not* necessarily any given point.

In this lecture, we only concern the problem of K-*center* clustering. Notice that the techniques to address K-means and K-median are somehow different from the techniques in K-center. For a reference, one can refer to [1].

2 Warm-up: A 2-Approximation Algorithm for k-Center

We demonstrate a 2-approximation algorithm in this section. Let us first clarify the goals and notions we are going to use. We use OPT to denote the (minimum) $\max_{p\in P} \|p - C^*_{(p)}\|$ by the real optimal K-center clusters $C^*_1, C^*_2, \cdots, C^*_K$. Our goal is to find some clusters $\tilde{C}_1, \cdots, \tilde{C}_K$ such that:

$$\max_{p\in P} \|p - \tilde{C}_{(p)}\| \leq 2 \cdot \text{OPT}$$

The algorithm follows a simple intuition: since we want to bound the maximum distance between each point and its clustering center, we can consider 'deal with the most stubborn one' at each round and continuously assign clustering centers to the points that are furthest from the 'frontier'. Moreover, notice that the any estimated clustering center $\tilde{C}_i$ should be in an OPT-range of some optimal cluster C_j. Thus, if we can show that any point covered by C_j can also be covered by $\tilde{C}_i$ or some even-better estimated clusters, we will get a 2-approximation algorithm.

Now we formally give the algorithm as the following:

Algorithm APPROX-K-CENTER: A 2-approximate K-center clustering algorithm.

1. Pick an arbitrary point p_i as $\tilde{C}_1$
2. For k=2:K
   (a) Pick the point with the furthest distance e.g. $\tilde{p} = \max_{p\in P} \|p - \tilde{C}_{(p)}\|_2$
   (b) Let $\tilde{p}$ be $\tilde{C}_k$

We now analyze the simple algorithm and show that it will return a 2-approximation of the K-center clustering in polynomial time.

Claim 1. *Algorithm APPROX-K-CENTER will return a 2-approximation of the K-center clustering.*

Proof. Suppose the underlying ground-truth clusters are C_1^*, C_2^*, $\cdots$, C_K^*. Then for any point p in C_i^*, we should have $\|p - C_i^*\|_2 \leq \cdot\text{OPT}$ (the 'radius').

Algorithm APPROX-K-CENTER can have two scenarios:

- APPROX-K-CENTER picks exactly one cluster $\tilde{C}_i$ from each C_j^*. In this case, any point p covered by C_j^* will be covered by $\tilde{C}_i$ with $\|p - \tilde{C}_i\|_2 \leq \|p - C_i^*\|_2 + \|C_i^* - \tilde{C}_i\|_2 \leq 2 \cdot \text{OPT}$.
- APPROX-K-CENTER picks two clusters from the same cluster. Suppose APPROX-K-CENTER picks $\tilde{C}_i$ and $\tilde{C}_k$ from C_j^*, and WLOG suppose $\tilde{C}_k$ is picked after $\tilde{C}_i$. Let us analyze the moment $\tilde{C}_k$ is picked: Since the algorithm always pick the furthest point, for any other point $p \in P/\{\tilde{C}_k\}$, we have

$$\begin{aligned}
\|p - \tilde{C}_{(p)}\|_2 &\leq \|\tilde{C}_k - \tilde{C}_{(\tilde{C}_k)}\|_2 \\
&\leq \|\tilde{C}_k - \tilde{C}_i\|_2 \\
&\leq \|\tilde{C}_k - C_j^*\|_2 + \|C_j^* - \tilde{C}_i\|_2 \\
&\leq 2 \cdot \text{OPT} && \text{(since both } \tilde{C}_k \text{ and } \tilde{C}_i \text{ are in } C_j^*\text{)}
\end{aligned}$$

□

Claim 2. *Algorithm APPROX-K-CENTER runs in polynomial time (e.g. $O(K \cdot n \cdot d)$, where n is the number of points and d is the number of dimensions).*

Proof. This is straightforward to see as the algorithm only run K iterations; And at each iteration, it takes $O(n \cdot d)$ time to compute the furthest point. □

We refer to the above algorithm as APPROX-K-CENTER in this notes as it will be used as a black-box in the remaining parts.

Remark. Notice that an important property of the above algorithm is it runs in *polynomial* time. The K-center clustering algorithm is itself *NP-hard* (at least as hard as NP); Moreover, it can be proven that unless $P = NP$, neither can we solve $(1 + \varepsilon)$-approximation ($\varepsilon \in (0, 1)$) for K-center clustering in polynomial time.

3 Streaming Clustering Algorithms: Coreset Approximation

We now turn to our main problem: steaming clustering. In this model, the points $p_1, p_2, \cdots, p_n \in \mathbb{R}^d$ are arriving in a streaming manner. We aim to design algorithms that can output K clustering centers by the end of the stream with a good approximation. In addition to the usual assumptions, in the steaming clustering model, we also assume the value of each dimension is a positive integer and is bounded. Formally, this can be denoted as:

$$p_i = \{1, 2, \cdots, \Delta\}^d$$

where Δ is the upper bound of the value on each dimension. Notice that in this way, we can store each data point in $O(d \log(\Delta))$ bits.

A naive way to solve this problem is to store all the points and perform APPROX-K-CENTER afterwards. The space complexity of this algorithm will therefore be $O(n \cdot d \cdot \log(n))$, which is polynomial in n and d, which is not sublinear. As usual, in the streaming algorithm setup, we are interested in finding algorithms with *sublinear space*.

Remark. It is a little slippery to talk about the *best possible approximation* factor in streaming clustering. The tricky part here is that in a streaming model, the time complexity is not explicitly bounded. Thus, even we can show a $(1+\varepsilon)$-approximation for K-center is unlikely to be possible in polynomial time, we may get $(1+\varepsilon)$-approximate algorithms in the streaming setting. Nevertheless, we do not pursue this direction in this lecture, and the algorithm we developed here are all of polynomial time.

3.1 Coreset Approximation

We are going to show one type of technique, namely coresets, that can help develop streaming clustering algorithm with sublinear space. The precise definition of a coreset is somehow tedious, but one can comprehend it as a 'sufficiently representative subset' of a given set. That is to say, a coreset will only contain a certain selected subset of bunch of data points, and the points are selected based on some metrics to preserve the characteristics. An illustration of the coresets can be shown as Figure 2.

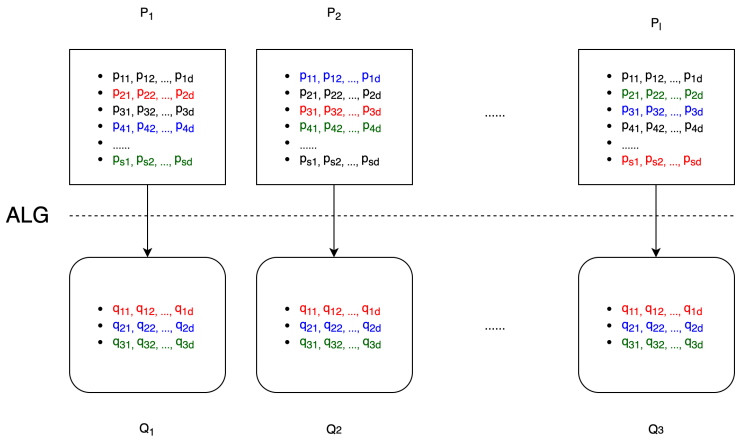

Figure 2: An example of coresets with size 3. The sets P_1, P_2, $\cdots$, P_ℓ contain different points; and the Red, Blue and Green points are picked by `ALG` to construct different coresets $Q_1, Q_2, \cdots, Q_\ell$.

In the sense of 'preserving characteristics', we define the metric as the 'stretch' on the clustering distances. Formally, we call a coreset $Q_1, Q_2, \cdots, Q_\ell$ as an *α-approximation coreset* if the clusters $\tilde{C}_1, \tilde{C}_2, \cdots, \tilde{C}_K$ based on $Q_1, Q_2, \cdots, Q_\ell$ is an α-approximation of the clusters $C_1^*, C_2^*, \cdots, C_K^*$ on points $P_1, P_2, \cdots, P_\ell$:

$$\max_{q \in \{Q_1, \cdots, Q_\ell\}} \|q - \tilde{C}_{(q)}\|_2 \leq \alpha \cdot \max_{p \in \{P_1, \cdots, P_\ell\}} \|p - C^*_{(p)}\|_2$$

It is worthy to mention that the idea of coreset is somehow similar to the linear sketch in the sparse/compressed sensing (Lecture 6). The difference is that in the linear sketch, we performed a linear project; while here we perform a subset selection.

For pedagogical reasons, suppose we are given an algorithm `ALG-1` that outputs coreset with size r and for now we do not care about guarantees on the distance preservation. We first design the following algorithm by using `ALG-1` as a blackbox:

Algorithm Coreset Streaming Clustering 1: A framework of streaming clustering with coresets.

1. Add arriving data points p to P_i until $\|P_i\|_0 = S$.
2. For every time $\|P_i\|_0 = S$:

(a) Construct a coreset Q_i with `ALG-1` and store.

(b) Discard P_i, set $i = i + 1$.

3. Get coreset $\{Q_1, Q_2, \cdots, Q_\ell\}$.

4. Run APPROX-K-CENTER on the coreset.

The algorithm looks extremely simple, and below we give the lemmas for its space complexity and correctness that will complete the framework of the algorithm.

Lemma 3. *The space complexity of the above algorithm is* $O(S + \frac{n}{S} \cdot r)$.

Proof. Through the algorithm, we maintain a single chunk of points P_i and $\frac{n}{S}$ coresets since the coreset will be constructed and saved every S points. We assume the size of every coreset is r, so the overall space complexity is $O(S + \frac{n}{S} \cdot r)$. □

Lemma 4. *Let points in the coreset* $q \in \cup_{i=1}^{\ell} Q_i$ *be centers for* K*-center clustering and denote them with* $\tilde{C}^Q_{(\cdot)}$*. If we have:*

$$\forall p \in \cup_{i=1}^{\ell} P_i, \quad \|p - \tilde{C}^Q_{(p)}\|_2 \leq 2 \cdot \mathrm{OPT}$$

then $\{Q_i\}_{i=1}^{\ell}$ *is a* 4*-approximate coreset.*

Proof. This statement means that if the coreset can cover the points with a 'radius' of $2 \cdot \mathrm{OPT}$, then the clustering center of the coresets ($\tilde{C}_{(\cdot)}$) will be able to cover all the points with a 'radius' of $4 \cdot \mathrm{OPT}$. To see this argument, we can first decompose the distance between p and $\tilde{C}_{(p)}$:

$$\begin{aligned} \|p - \tilde{C}_{(p)}\| &\leq \|p - \tilde{C}^Q_{(p)}\| + \|\tilde{C}^Q_{(p)} - \tilde{C}_{(p)}\| \\ &\leq 2 \cdot \mathrm{OPT} + \|\tilde{C}^Q_{(p)} - \tilde{C}_{(p)}\| \qquad \text{(By assumption)} \end{aligned}$$

Now the only work is to prove $\|\tilde{C}^Q_{(p)} - \tilde{C}_{(p)}\| \leq 2 \cdot \mathrm{OPT}$. Notice that the $\tilde{C}^Q_{(p)}$ are actually the q points and $\tilde{C}_{(p)}$ is their center, so we can directly invoke Claim 1 to conclude the proof. □

3.2 APPROX-K-CENTER for the Coresets

Upon proving Lemma 4, keen readers might have already noticed a way to implement the coreset construction to have a 4-approximation streaming clustering algorithm – we can just run APPROX-K-CENTER on P_i and pick the clustering centers as the coreset. One may also wonder if we can simply invoke Claim 1 again to prove $\|p - \tilde{C}^Q_{(p)}\|_2 \leq 2 \cdot \mathrm{OPT}$. Unfortunately, it turns out we need to prove this separately, as the direct guarantee is only 'local' in terms of P_i instead of 'global'.

Claim 5. *If we use APPROX-K-CENTER as the* `ALG-1` *in* ***Algorithm Coreset Streaming Clustering 1****, then there is:*

$$\|p - \tilde{C}^Q_{(p)}\|_2 \leq 2 \cdot \mathrm{OPT}$$

Proof. We can again use the same strategy of the proof in Claim 1 with some minor modifications. For each true cluster C^*_k, the points in the coreset has two scenarios:

- There exists some point $q = \tilde{C}^Q_j \in \cup_{i=1}^{\ell} Q_i$ such that $q \in C^*_k$. In this case, any point p covered by C^*_k will be covered by $\tilde{C}^Q_j$ with $\|p - \tilde{C}^Q_j\|_2 \leq \|p - C^*_k\|_2 + \|C^*_k - \tilde{C}^Q_j\|_2 \leq 2 \cdot \mathrm{OPT}$.

- For all $q = \tilde{C}_j^Q \in \cup_{i=1}^{\ell} Q_i$, we have $q \notin C_k^*$. Since we are picking K qs each time, if for C_k^* no q value is within its cluster, then it means we have two qs in the same cluster, thus the proof can be done in the same way of the case 2 of Claim 1.

□

Taking together the above results, now we can get the following theorem:

Theorem 6 (4-approximation streaming clustering)**.** *There exists a 4-approximation algorithm for streaming clustering with points $p_1, \cdots, p_n \in \{1, 2, \cdots, \Delta\}^d$ with the space complexity of $O(\sqrt{nk} \cdot d \cdot \log(n))$ bits.*

Proof. We are going to use the conclusions in Lemmas 3 and 4 and Claim 5. Notice now with the APPROX-K-CENTER algorithm as coreset construction, we can get a 4-approximation algorithm already. Also consider the space complexity of $O(S + \frac{n}{S} \cdot r)$, and now $r = K$. If we let $S = \sqrt{nK}$, then this bound will become $O(\sqrt{nk})$. We also remark that with the arithmetic inequality, this is the best space complexity we can get based on this algorithm □

4 More Efficient Streaming Clustering Algorithms

We now have a $O(\sqrt{nk})$ space complexity algorithm. This is a big improvement with respect to the naive $\text{poly}(n, k, d)$ algorithm; However, in the regime of streaming models, this type of space complexity is not very satisfactory. In this section, we show two strategies to further improve the space complexity.

4.1 Multi-level Coresets – Trade Approximation for Efficiency

In the previous section, we have shown that with the APPROX-K-CENTER algorithm to construct coresets, one can implement another APPROX-K-CENTER over the obtained coresets. Now, we can also apply the coreset to multiple levels – namely, upon we have collected S points, we will run APPROX-K-CENTER to get a coreset; we will collect multiple coresets, and upon we have collected S coresets, we run another APPROX-K-CENTER to get a 'second-level coresets'. We continue this process until it reaches the level we want.

For 2 levels of coresets, we need to maintain $O(S)$ points on the first level, $O(K \cdot S)$ first-level coresets, and $O(\frac{n}{S^2} \cdot K)$ coresets for the final clustering. The summation of the three terms will meet its minimum when $S = O(n^{\frac{1}{3}})$, where the space complexity will become $O(K \cdot n^{\frac{1}{3}})$. Notice that by performing such type of multi-level coreset, we also traded the approximation factor – the distance bound will be doubled upon we add the new layer.

In general, this idea can be formulated as the following statement.

Proposition 7 (Multi-level coreset Streaming Clustering)**.** *With the APPROX-K-CENTER algorithm as the coreset construction algorithm, there exists an 4^L-approximate algorithm with space complexity of $O(n^{\frac{1}{L+1}} \cdot K \cdot d \cdot \log(\Delta))$ bits , where L is the number of coresets constructions.*

We omit the proof of this proposition due to the limit of space and the similarities with the proof of `Algorithm Coreset Streaming Clustering 1`.

4.2 K-Center Testing for Streaming Clustering

So far, apart from bounding the number of bits for each element, we have not leverage the condition of $p \in \{1, 2, \cdots, \Delta\}^d$. Recall what we did in Lecture 7: in that lecture, we used a geometric search to approximate the number of distinct elements in a stream. The lesson we learned is, if the quantity of the desired value is discrete, bounded, and an approximation of the quantity is sufficient, we can leverage

the power of geometric testing and output the answer that 'crosses the threshold' between two contrasting outputs. We now give another algorithm based on the above idea, and it has a much better approximation factor and space complexity.

Consider the following problem as our test:

Problem 1 (K-center Testing)**.** Given a stream of points $p_1, p_2, \cdots, p_n \in \mathbb{R}^d$ and parameter $\tilde{T}$. Output K-center clustering results as:

1. If $\max_p \|p - \tilde{C}_{(p)}\| \leq 2 \cdot \tilde{T}$, then return the clustering centers $\{\tilde{C}_i\}_{i=1}^K$.
2. Otherwise, return OPT $> \tilde{T}$.

Again, for pedagogical reasons, let us first assume we have an algorithm `ALG-2` such that it will correctly solve the K-center Testing problem. Now, we can use `ALG-2` to design an algorithm that solves the K-center clustering problem:

Algorithm Coreset Streaming Clustering 2: A framework of streaming clustering with geometric tests.

1. Let the parameter list $T = \{1, (1+\varepsilon), (1+\varepsilon)^2, \cdots, \log_{(1+\varepsilon)}(\Delta)\}$.
2. Run `ALG-2` in parallel for each $\tilde{T} \in T$ for arriving stream $p_1, \cdots, p_n$.
3. Output the smallest answer with clustering results (instead of 'OPT $> \tilde{T}$').

Lemma 8. *Suppose `ALG-2` will solve the K-center Testing problem with the space complexity b bits, then the above algorithm gives a $2 \cdot (1+\varepsilon)$-approximation of streaming K-center clustering with $O(\frac{\log(\Delta)}{\varepsilon} \cdot b)$ bits space complexity.*

Proof. The space complexity is straightforward to prove: since we have $O(\frac{\log(\Delta)}{\varepsilon})$ rounds at most, and each round can be solved with space complexity b, if we run each round *in parallel*, the total space complexity will be $O(\frac{\log(\Delta)}{\varepsilon} \cdot b)$ bits.

The proof of correctness goes as follows: suppose the $(i+1)$-th is the last index for OPT $> \tilde{T}$. That is, for $\tilde{T} = (1+\varepsilon)^i$, `ALG-2` returns OPT $> \tilde{T}$; and for $\tilde{T} = (1+\varepsilon)^{i+1}$, `ALG-2` returns $\{\tilde{C}_i\}_{i=1}^K$ such that c. Also, notice that the algorithm returns correctly at $\tilde{T} = (1+\varepsilon)^i$, thus OPT $> (1+\varepsilon)^i$. Taking the above together, we have:

$$\begin{aligned}\max_{p \in P} \|p - \tilde{C}_{(p)}\| &\leq 2 \cdot \tilde{T} = (1+\varepsilon)^{i+1} \\ &= 2 \cdot (1+\varepsilon) \cdot (1+\varepsilon)^i \\ &\leq 2 \cdot (1+\varepsilon) \cdot \text{OPT}\end{aligned}$$

which concludes the proof. □

An algorithm solves K-center Testing with $K \cdot d \cdot \log(n)$ space

Upon now, we are ready for everything other than the `ALG-2` to actually solve the problem. We now give `ALG-2` as the following:

ALG-2: Correctly answer the K-center Testing problem.

1. Let $\mathcal{C} = \emptyset$.
2. For each arriving p_i, if $\|p_i - C_{(p)}\| > 2 \cdot \tilde{T}, \quad \forall C \in \mathcal{C}$, then add p_i to $\mathcal{C}$ (e.g. $\mathcal{C} = \mathcal{C} \cup \{p_i\}$).
3. If at any point, $|\mathcal{C}| > k$, then output OPT $> \tilde{T}$.
4. Otherwise, return $\mathcal{C}$.

Claim 9. *If* OPT $\leq \tilde{T}$*, then* `ALG-2` *returns* $\{\tilde{C}_i\}_{i=1}^K$ *with* $\max_{p \in P} \|p - \tilde{C}_{(p)}\| \leq 2\tilde{T}$*; otherwise,* `ALG-2` *returns* OPT $> \tilde{T}$.

Proof. We can easily prove that if the algorithm returns, it must have $\max_{p \in P} \|p - \tilde{C}_{(p)}\| \leq 2\tilde{T}$ as the Line 2 specified. Thus, if we can prove under the algorithm, $|\mathcal{C}| > k$ only if OPT $> \tilde{T}$, then we are done with the proof.

Consider prove by contradiction. Suppose now OPT $\leq \tilde{T}$ and we run into $|\mathcal{C}| > k$ at some point. That means we have $(K+1)$ points with mutual distance greater than $2\tilde{T}$. Now since we assume OPT $\leq \tilde{T}$, then no ground-truth cluster C_i^* can have more than 1 point picked, as otherwise we will have at most $2\tilde{T}$ distance. However, we picked $(K+1)$ point with only K clusters, which violates the pigeonhole principle. Thus, if $|\mathcal{C}| > k$ there must be OPT $> \tilde{T}$. □

Claim 10. `ALG-2` *runs with the space complexity of* $O(K \cdot d \cdot \log(\Delta))$ *bits.*

Proof. This is straightforward to show as we store at most K points, and each point only takes $O(d \log(\Delta))$ bits to store. Also, we remark that the time for this algorithm is polynomial. □

Putting together what we have shown, we can conclude the section with the following formal theorem:

Theorem 11 ($2(1+\varepsilon)$-approximation streaming clustering)**.** *There exists a* $2 \cdot (1+\varepsilon)$*-approximation algorithm for streaming clustering with points* $p_1, \cdots, p_n \in \{1, 2, \cdots, \Delta\}^d$ *with the space complexity of* $O(K \cdot d \cdot \frac{\log^2(\Delta)}{\varepsilon})$ *bits.*

The proof of correctness simply follows the combination of Lemma 8 and Claim 9. The proof of space complexity can be done by plugging Claim 10 into Lemma 8.

References

[1] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005. 2