Improved Guarantees for k-means++ and k-means++ Parallel

Konstantin Makarychev, Aravind Reddy, and Liren Shan

Department of Computer Science
Northwestern University
Evanston, IL, USA

Abstract

In this paper, we study k-means++ and k-means$\|$, the two most popular algorithms for the classic k-means clustering problem. We provide novel analyses and show improved approximation and bi-criteria approximation guarantees for k-means++ and k-means$\|$. Our results give a better theoretical justification for why these algorithms perform extremely well in practice. We also propose a new variant of k-means$\|$ algorithm (Exponential Race k-means++) that has the same approximation guarantees as k-means++.

1 Introduction

k-means clustering is one of the most commonly encountered unsupervised learning problems. Given a set of n data points in Euclidean space, our goal is to partition them into k clusters (each characterized by a center), such that the sum of squares of distances of data points to their nearest centers is minimized. The most popular heuristic for solving this problem is Lloyd's algorithm (Lloyd, 1982), often referred to simply as "the k-means algorithm".

Lloyd's algorithm uses iterative improvements to find a locally optimal k-means clustering. The performance of Lloyd's algorithm crucially depends on the quality of the initial clustering, which is defined by the initial set of centers, called a *seed*. Arthur and Vassilvitskii (2007) and Ostrovsky, Rabani, Schulman, and Swamy (2006) developed an elegant randomized seeding algorithm, known as the k-means++ algorithm. It works by choosing the first center uniformly at random from the data set and then choosing the subsequent $k-1$ centers by randomly sampling a single point in each round with the sampling probability of every point proportional to its current cost. That is, the probability of choosing any data point x is proportional to the squared distance to its closest already chosen center. This squared distance is often denoted by $D^2(x)$. Arthur and Vassilvitskii (2007) proved that the expected cost of the initial clustering obtained by k-means++ is at most $8\,(\ln k + 2)$ times the cost of the optimal clustering i.e., k-means++ gives an $8\,(\ln k + 2)$-approximation for the k-means problem. They also provided a family of k-means instances for which the approximation factor of k-means++ is $2\ln k$ and thus showed that their analysis of k-means++ is almost tight.

Due to its speed, simplicity, and good empirical performance, k-means++ is the most widely used algorithm for k-means clustering. It is employed by such machine learning libraries as Apache

The conference version of this paper will appear in the proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Author order is alphabetical.

Spark MLlib, Google BigQuery, IBM SPSS, Intel DAAL, and Microsoft ML.NET. In addition to k-means++, these libraries implement a scalable variant of k-means++ called k-means$\|$ (read "k-means parallel") designed by Bahmani, Moseley, Vattani, Kumar, and Vassilvitskii (2012). Somewhat surprisingly, k-means$\|$ not only works better in parallel than k-means++ but also slightly outperforms k-means++ in practice in the single machine setting (see Bahmani et al. (2012) and Figure 1 below). However, theoretical guarantees for k-means$\|$ are substantially weaker than for k-means++.

The k-means$\|$ algorithm makes T passes over the data set (usually $T = 5$). In every round, it independently draws approximately $\ell = \Theta(k)$ random centers according to the D^2 distribution. After each round it recomputes the distances to the closest chosen centers and updates $D^2(x)$ for all x in the data set. Thus, after T rounds, k-means$\|$ chooses approximately $T\ell$ centers. It then selects k centers among $T\ell$ centers using k-means++.

Our contributions. In this paper, we improve the theoretical guarantees for k-means++, k-means$\|$, and Bi-Criteria k-means++ (which we define below).

First, we show that the expected cost of the solution output by k-means++ is at most $5(\ln k+2)$ times the optimal solution's cost. This improves upon the bound of $8(\ln k+2)$ shown by Arthur and Vassilvitskii (2007) and directly improves the approximation factors for several algorithms which use k-means++ as a subroutine like Local Search k-means++ (Lattanzi and Sohler, 2019). To obtain this result, we give a refined analysis of the expected cost of *covered clusters* (see Lemma 3.2 in Arthur and Vassilvitskii (2007) and Lemma 4.1 in this paper). We also show that our new bound on the expected cost of *covered clusters* is tight (see Lemma C.1).

Then, we address the question of why the observed performance of k-means$\|$ is better than the performance of k-means++. There are two possible explanations for this fact. (1) This may be the case because k-means$\|$ picks k centers in two stages. At the first stage, it samples $\ell T \geq k$ centers. At the second stage, it prunes centers and chooses k centers among ℓT centers using k-means++. (2) This may also be the case because k-means$\|$ updates the distribution function $D^2(x)$ once in every round. That is, it recomputes $D^2(x)$ once for every ℓ chosen centers, while k-means++ recomputes $D^2(x)$ every time it chooses a center. In this paper, we empirically demonstrate that the first explanation is correct. First, we noticed that k-means$\|$ for $\ell \cdot T = k$ is almost identical with k-means++ (see Appendix A). Second, we compare k-means$\|$ with another algorithm which we call Bi-Criteria k-means++ with Pruning. This algorithm also works in two stages: At the Bi-Criteria k-means++ stage, it chooses $k + \Delta$ centers in the data set using k-means++. Then, at the Pruning stage, it picks k centers among the $k + \Delta$ centers selected at the first stage again using k-means++. Our experiments on the standard data sets BioTest from KDD-Cup 2004 (Elber, 2004) and COVTYPE from the UCI ML repository (Dua and Graff, 2017) show that the performance of k-means$\|$ and Bi-Criteria k-means++ with Pruning are essentially identical (see Figures 1 and Appendix A).

These results lead to another interesting question: How good are k-means++ and k-means$\|$ algorithms that sample $k+\Delta$ instead of k centers? The idea of oversampling using k-means++ was studied earlier in the literature under the name of *bi-criteria approximation*. Aggarwal, Deshpande, and Kannan (2009) showed that with constant probability, sampling $k + \Delta$ centers by k-means++ provides a constant-factor approximation if $\Delta \geq \beta k$ for some constant $\beta > 0$. Wei (2016) improved on this result by showing an expected approximation ratio of $8(1 + 1.618k/\Delta)$. Note that for bi-criteria algorithms we compare the expected cost of the clustering with $k + \Delta$ centers they produce and the cost of the optimal clustering with exactly k centers.

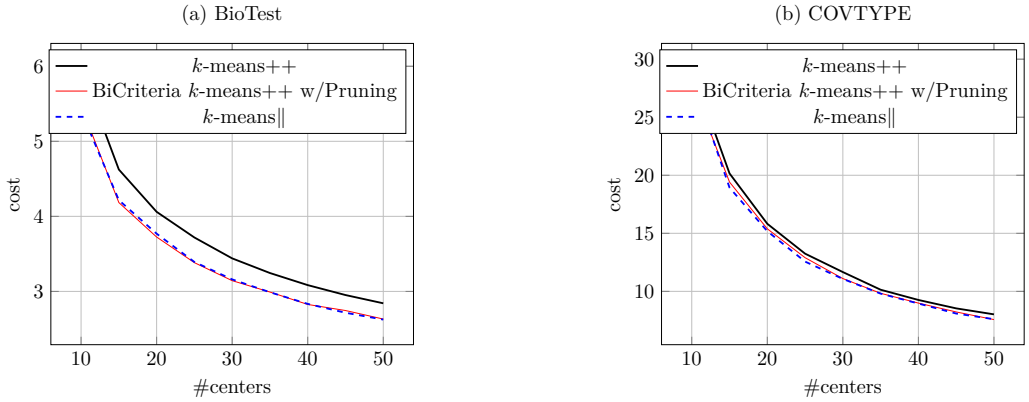


Figure 1: Performance of k-means++, k-means‖, and Bi-Criteria k-means++ with pruning on the BioTest and COVTYPE datasets. For $k = 10, 15, \cdots, 50$, we ran these algorithms for 50 iterations and took their average. We normalized the clustering costs by dividing them by $\text{cost}_{1000}(\mathbf{X})$.

In this paper, we show that the expected bi-criteria approximation ratio for k-means++ with Δ additional centers is at most the minimum of two bounds:

$$\text{(A) } 5\left(2 + \frac{1}{2e} + \ln\frac{2k}{\Delta}\right) \text{ for } 1 \le \Delta \le 2k; \text{ and (B) } 5\left(1 + \frac{k}{e\,(\Delta - 1)}\right) \text{ for } \Delta \ge 1$$

Both bounds are better than the bound by Wei (2016). The improvement is especially noticeable for small values of Δ. More specifically, when the number of additional centers is $\Delta = k/\log k$, our approximation guarantee is $O(\log\log k)$ while Wei (2016) gives an $O(\log k)$ approximation.

We believe that our results for small values of Δ provide an additional explanation for why k-means++ works so well in practice. Consider a data scientist who wants to cluster a data set $\mathbf{X}$ with k^* *true clusters* (i.e. k^* latent groups). Since she does not know the actual value of k^*, she uses the *elbow method* (Boehmke and Greenwell, 2019) or some other heuristic to find k. Our results indicate that if she chooses slightly more number of clusters (for instance, $1.05k^*$), then she will get a constant bi-criteria approximation to the optimal clustering.

We also note that our bounds on the approximation factor smoothly transition from the regular ($\Delta = 0$) to bi-criteria ($\Delta > 0$) regime. We complement our analysis with an almost matching lower bound of $\Theta(\log(k/\Delta))$ on the approximation factor of k-means for $\Delta \le k$ (see Appendix C).

We then analyze Bi-Criteria k-means‖ algorithm, the variant of k-means‖ that does not prune centers at the second stage. In their original paper, Bahmani, Moseley, Vattani, Kumar, and Vassilvitskii (2012) showed that the expected cost of the solution for k-means‖ with T rounds and oversampling parameter ℓ is at most:

$$\frac{16}{1-\alpha}\text{OPT}_k(\mathbf{X}) + \left(\frac{1+\alpha}{2}\right)^T \text{OPT}_1(\mathbf{X}),$$

where $\alpha = \exp\left(-\left(1 - e^{-\ell/(2k)}\right)\right)$; $\text{OPT}_k(\mathbf{X})$ is the cost of the optimal k-means clustering of $\mathbf{X}$; $\text{OPT}_1(\mathbf{X})$ is the cost of the optimal clustering of X with 1 center (see Section 2 for details). We note that $\text{OPT}_1(\mathbf{X}) \gg \text{OPT}_k(\mathbf{X})$. For $\ell = k$, this result gives a bound of $\approx 49\,\text{OPT}_k(\mathbf{X}) + 0.83^T\text{OPT}_1(\mathbf{X})$. Bachem, Lucic, and Krause (2017) improved the approximation guarantee for

$\ell \geq k$ to

$$26\mathrm{OPT}_k(\mathbf{X}) + 2\Big(\frac{k}{e\ell}\Big)^T \mathrm{OPT}_1(\mathbf{X}).$$

In this work, we improve this bound for $\ell \geq k$ and also obtain a better bound for $\ell < k$. For $\ell \geq k$, we show that the cost of k-means$\|$ without pruning is at most

$$8\mathrm{OPT}_k(\mathbf{X}) + 2\Big(\frac{k}{e\ell}\Big)^T \mathrm{OPT}_1(\mathbf{X}).$$

For $\ell < k$, we give a bound of

$$\frac{5}{1 - e^{-\frac{\ell}{k}}}\,\mathrm{OPT}_k(\mathbf{X}) + 2\left(e^{-\frac{\ell}{k}}\right)^T \mathrm{OPT}_1(\mathbf{X})$$

Finally, we give a new parallel variant of the k-means++ algorithm, which we call *Exponential Race k-means++* (k-means++$_{\mathrm{ER}}$). This algorithm is similar to k-means$\|$. In each round, it also selects ℓ candidate centers in parallel (some of which may be dropped later) making one pass over the data set. However, after T rounds, it returns exactly k centers. The probability distribution of these centers is identical to the distribution of centers output by k-means++. The expected number of rounds is bounded as follows:

$$O\Big(\frac{k}{\ell} + \log \frac{\mathrm{OPT}_1(\mathbf{X})}{\mathrm{OPT}_k(\mathbf{X})}\Big).$$

This algorithm offers a unifying view on k-means++ and k-means$\|$. We describe it in Section 7.

Other related work. Dasgupta (2008) and Aloise, Deshpande, Hansen, and Popat (2009) showed that k-means problem is NP-hard. Awasthi, Charikar, Krishnaswamy, and Sinop (2015) proved that it is also NP-hard to approximate k-means objective within a factor of $(1 + \varepsilon)$ for some constant $\varepsilon > 0$ (see also Lee, Schmidt, and Wright (2017)). We also mention that k-means was studied not only for Euclidean spaces but also for arbitrary metric spaces.

There are several known *constant* factor approximation algorithms for the k-means problem. Kanungo, Mount, Netanyahu, Piatko, Silverman, and Wu (2004) gave a $9 + \varepsilon$ approximation local search algorithm. Ahmadian, Norouzi-Fard, Svensson, and Ward (2019) proposed a primal-dual algorithm with an approximation factor of 6.357. This is the best known approximation for k-means. Makarychev, Makarychev, Sviridenko, and Ward (2016) gave constant-factor bi-criteria approximation algorithms based on linear programming and local search. Note that although these algorithms run in polynomial time, they do not scale well to massive data sets. Lattanzi and Sohler (2019) provided a constant factor approximation by combining the local search idea with the k-means++ algorithm. Choo, Grunau, Portmann, and Rozhoň (2020) further improved upon this result by reducing the number of local search steps needed from $O(k \log\log k)$ to $O(k)$.

Independently and concurrently to our work, Rozhoň (2020) gave an interesting analysis for k-means$\|$ by viewing it as a *balls into bins* problem and showed that $O(\log n / \log\log n)$ rounds suffice to give a constant approximation with high probability.

Acknowledgments. We would like to thank all the reviewers for their helpful comments. Konstantin Makarychev, Aravind Reddy, and Liren Shan were supported in part by NSF grants CCF-1955351 and HDR TRIPODS CCF-1934931. Aravind Reddy was also supported in part by NSF CCF-1637585.

2 Preliminaries

Given a set of points $\mathbf{X} = \{x_1, x_2, \cdots, x_n\} \subseteq \mathbb{R}^d$ and an integer $k \geq 1$, the k-means clustering problem is to find a set C of k centers in $\mathbb{R}^d$ to minimize

$$\text{cost}(\mathbf{X}, C) \coloneqq \sum_{x \in \mathbf{X}} \min_{c \in C} \|x - c\|^2.$$

For any integer $i \geq 1$, let us define $\text{OPT}_i(\mathbf{X}) \coloneqq \min_{|C|=i} \text{cost}(\mathbf{X}, C)$. Thus, $\text{OPT}_k(\mathbf{X})$ refers to the cost of the optimal solution for the k-means problem. Let C^* denote a set of optimal centers. We use $\{P_i\}_{i=1}^k$ to denote the clusters induced by the center set C^*.

For any $\mathbf{Y} \subseteq \mathbf{X}$, the cost of $\mathbf{Y}$ with center set C, denoted by $\text{cost}(\mathbf{Y}, C) = \sum_{x \in \mathbf{Y}} \min_{c \in C} \|x - c\|^2$. The optimal cost for subset $\mathbf{Y}$ with i centers is $\text{OPT}_i(\mathbf{Y})$. Let $\mu = \sum_{x \in \mathbf{Y}} x / |\mathbf{Y}|$ be the *centroid* of the cluster $\mathbf{Y}$. Then, we have the following closed form expression for the optimal cost of $\mathbf{Y}$ with one center (see Appendix B for proof),

$$\text{OPT}_1(\mathbf{Y}) = \sum_{x \in \mathbf{Y}} \|x - \mu\|^2 = \frac{\sum_{(x,y) \in \mathbf{Y} \times \mathbf{Y}} \|x - y\|^2}{2\,|\mathbf{Y}|}. \tag{1}$$

k-means++ seeding: The k-means++ algorithm samples the first center uniformly at random from the given points and then samples $k - 1$ centers sequentially from the given points with probability of each point being sampled proportional to its cost i.e. $\text{cost}(x, C)/\text{cost}(\mathbf{X}, C)$.

Algorithm 1 k-means++ seeding

1: Sample a point c uniformly at random from $\mathbf{X}$ and set $C_1 = \{c\}$.
2: **for** $t = 2$ **to** k **do**
3: Sample $x \in \mathbf{X}$ w.p. $\text{cost}(x, C_t)/\text{cost}(\mathbf{X}, C_t)$.
4: $C_t = C_{t-1} \cup \{x\}$.
5: **end for**
6: **Return** C_k

k-means$\|$ and k-means$\|_{\text{Pois}}$ seeding: In the k-means$\|$ algorithm, the first center is chosen uniformly at random from $\mathbf{X}$. But after that, at each round, the algorithm samples each point independently with probability $\min\{\ell \cdot \text{cost}(x, C)/\text{cost}(\mathbf{X}, C), 1\}$ where ℓ is the *oversampling parameter* chosen by the user and it usually lies between $0.1k$ and $10k$. The algorithm runs for T rounds (where T is also a parameter chosen by the user) and samples around ℓT points, which is usually strictly larger than k. This oversampled set is then weighted using the original data set $\mathbf{X}$ and a weighted version of k-means++ is run on this set to get the final k-centers. We only focus on the stage in which we get the oversampled set because the guarantees for the second stage come directly from k-means++.

For the sake of analysis, we also consider a different implementation of k-means$\|$, which we call k-means$\|_{\text{Pois}}$ (Algorithm 3). This algorithm differs from k-means$\|$ in that each point is sampled independently with probability $1 - \exp(-\ell \cdot \text{cost}(x, C)/\text{cost}(\mathbf{X}, C))$ rather than $\min\{\ell \cdot \text{cost}(x, C)/\text{cost}(\mathbf{X}, C), 1\}$. In practice, there is essentially no difference between k-means$\|$ and k-means$\|_{\text{Pois}}$, since $\ell \cdot \text{cost}(x, C)/\text{cost}(\mathbf{X}, C)$ is a very small number for all x and thus the sampling probabilities for k-means$\|$ and k-means$\|_{\text{Pois}}$ are almost equal.

Algorithm 2 k-means$\|$ seeding

1: Sample a point c uniformly from $\mathbf{X}$ and set $C_1 = \{c\}$
2: **for** $t = 1$ **to** T **do**
3: Sample each point x into C' independently w.p. $\min\{1, \lambda_t(x)\}$ where $\lambda_t(x) = \ell \cdot \text{cost}(x, C_t)/\text{cost}(\mathbf{X}, C_t)$
4: Let $C_{t+1} = C_t \cup C'$.
5: **end for**

Algorithm 3 k-means$\|_{\text{Pois}}$ seeding

1: Sample a point c uniformly from $\mathbf{X}$ and set $C_1 = \{c\}$
2: **for** $t = 1$ **to** T **do**
3: Sample each point x into C' independently w.p. $1 - e^{-\lambda_t(x)}$ where $\lambda_t(x) = \ell \cdot \text{cost}(x, C_t)/\text{cost}(\mathbf{X}, C_t)$
4: Let $C_{t+1} = C_t \cup C'$.
5: **end for**

In the rest of the paper, we focus only on the *seeding* step of k-means++, k-means$\|$, and k-means$\|_{\text{Pois}}$ and ignore Lloyd's iterations as the approximation guarantees for these algorithms come entirely from the seeding step.

3 General framework

In this section, we describe a general framework we use to analyze k-means++ and k-means$\|_{\text{Pois}}$. Consider k-means++ or k-means$\|_{\text{Pois}}$ algorithm. Let C_t be the set of centers chosen by this algorithm after step t. For the sake of analysis, we assume that C_t is an ordered set or list of centers, and the order of centers in C_t is the same as the order in which our algorithm chooses these centers. We explain how to order centers in k-means$\|_{\text{Pois}}$ algorithm in Section 6. We denote by T the stopping time of the algorithm. Observe that after step t of the algorithm, the probabilities of choosing a new center in k-means++ or a batch of new centers in k-means$\|_{\text{Pois}}$ are defined by the current costs of points in $\mathbf{X}$ which, in turn, are completely determined by the current set of centers C_t. Thus, the states of the algorithm form a Markov chain.

In our analysis, we fix the optimal clustering $\mathcal{P} = \{P_1, \ldots, P_k\}$ (if this clustering is not unique, we pick an arbitrary optimal clustering). The optimal cost of each cluster P_i is $\text{OPT}_1(P_i)$ and the optimal cost of the entire clustering is $\text{OPT}_k(\mathbf{X}) = \sum_{i=1}^k \text{OPT}_1(P_i)$.

Following the notation in Arthur and Vassilvitskii (2007), we say that a cluster P_i is *hit* or *covered* by a set of centers C if $C \cap P_i \neq \varnothing$; otherwise, we say that P_i is *not hit* or *uncovered*. We split the cost of each cluster P_i into two components which we call the covered and uncovered costs of P_i. For a given set of centers C,

$$\text{The covered or hit cost of } P_i, \qquad H(P_i, C) := \begin{cases} \text{cost}(P_i, C), & \text{if } P_i \text{ is covered by } C \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{The uncovered cost of } P_i, \qquad U(P_i, C) := \begin{cases} 0, & \text{if } P_i \text{ is covered by } C \\ \text{cost}(P_i, C), & \text{otherwise.} \end{cases}$$

Let $H(\mathbf{X}, C) = \sum_{i=1}^k H(P_i, C)$ and $U(\mathbf{X}, C) = \sum_{i=1}^k U(P_i, C)$. Then,

$$\text{cost}(\mathbf{X}, C) = H(\mathbf{X}, C) + U(\mathbf{X}, C).$$

For the sake of brevity, we define $\text{cost}_t(\mathbf{Y}) := \text{cost}(\mathbf{Y}, C_t)$ for any $\mathbf{Y} \subseteq \mathbf{X}$, $H_t(P_i) := H(P_i, C_t)$, and $U_t(P_i) := U(P_i, C_t)$. In Section 4, we show that for any t, we have $\mathbb{E}[H_t(\mathbf{X})] \leq 5\text{OPT}_k(\mathbf{X})$, which is an improvement over the bound of $8\text{OPT}_k(\mathbf{X})$ given by Arthur and Vassilvitskii (2007). Then, in

Sections 5 and 6, we analyze the expected uncovered cost $U(\mathbf{X}, C_T)$ for k-means++ and k-means$\|$ algorithms.

Consider a center c in C. We say that c is a *miss* if another center c' covers the same cluster in $\mathcal{P}$ as c, and c' appears before c in the ordered set C. We denote the number of misses in C by $M(C)$ and the the number of clusters in $\mathcal{P}$ not covered by centers in C by $K(C)$.

Observe that the stochastic processes $U_t(P_i)$ with discrete time t are non-increasing since the algorithm never removes centers from the set C_t and therefore the distance from any point x to C_t never increases. Similarly, the processes $H_t(P_i)$ are non-increasing after the step t_i when P_i is covered first time. In this paper, we sometimes use a proxy $\widetilde{H}_t(P_i)$ for $H_t(P_i)$, which we define as follows. If P_i is covered by C_t, then $\widetilde{H}_t(P_i) = H_{t_i}(P_i)$, where $t_i \le t$ is the first time when P_i is covered by C_t. If P_i is not covered by C_t, then $\widetilde{H}_t(P_i) = 5\mathrm{OPT}_1(P_i)$. It is easy to see that $H_t(P_i) \le \widetilde{H}_{t'}(P_i)$ for all $t \le t'$. In Section 4, we also show that $\widetilde{H}_t(P_i)$ is a supermartingale i.e., $\mathbb{E}[\widetilde{H}_{t'}(P_i) \mid C_t] \le \widetilde{H}_t(P_i)$ for all $t \le t'$.

4 Bound on the cost of covered clusters

In this section, we improve the bound by Arthur and Vassilvitskii (2007) on the expected cost of a covered cluster in k-means++. Our bound also works for k-means$\|_{\mathrm{Pois}}$ algorithm. Pick an arbitrary cluster P_i in the optimal solution $\mathcal{P} = \{P_1, \dots, P_k\}$ and consider an arbitrary state $C_t = \{c_1, \dots, c_N\}$ of the k-means++ or k-means$\|_{\mathrm{Pois}}$ algorithm. Let D_{t+1} be the set of new centers the algorithm adds to C_t at step t (for k-means++, D_{t+1} contains only one center). Suppose now that centers in D_{t+1} cover P_i i.e. $D_{t+1} \cap P_i \neq \varnothing$. We show that the expected cost of cluster P_i after step $(t+1)$ conditioned on the event $\{D_{t+1} \cap P_i \neq \varnothing\}$ and the current state of the algorithm C_t is upper bounded by $5\mathrm{OPT}_1(P_i)$ i.e.

$$\mathbb{E}\left[\mathrm{cost}(P_i, C_{t+1}) \mid C_t, \{D_{t+1} \cap P_i \neq \varnothing\}\right] \le 5\mathrm{OPT}_1(P_i). \tag{2}$$

We now prove the main lemma.

Lemma 4.1. *Consider an arbitrary set of centers $C = \{c_1, \dots, c_N\} \subseteq \mathbb{R}^d$ and an arbitrary set $P \subseteq \mathbf{X}$. Pick a random point c in P with probability $\Pr(c = x) = \mathrm{cost}(x, C)/\mathrm{cost}(P, C)$. Let $C' = C \cup \{c\}$. Then, $\mathbb{E}_c\left[\mathrm{cost}(P, C')\right] \le 5\mathrm{OPT}_1(P)$.*

Remarks: Lemma 3.2 in the paper by Arthur and Vassilvitskii (2007) gives a bound of $8\mathrm{OPT}_1(P)$. We also show in Appendix C that our bound is tight (see Lemma C.1).

Proof. The cost of any point y after picking center c equals the squared distance from y to the set of centers $C' = C \cup \{c\}$, which in turn equals $\min\{\mathrm{cost}(y, C), \|y - c\|^2\}$. Thus, if a point $x \in P$ is chosen as a center, then the cost of point y equals $\min\{\mathrm{cost}(y, C), \|x - y\|^2\}$. Since $\Pr(c = x) = \mathrm{cost}(x, C)/\mathrm{cost}(P, C)$, we have

$$\mathbb{E}_c\left[\mathrm{cost}(P, C')\right] = \sum_{\substack{x \in P \\ y \in P}} \frac{\mathrm{cost}(x, C)}{\mathrm{cost}(P, C)} \cdot \min\{\mathrm{cost}(y, C), \|x - y\|^2\}.$$

We write the right hand side in a symmetric form with respect to x and y. To this end, we define function f as follows:

$$f(x, y) = \mathrm{cost}(x, C) \cdot \min\left\{\|x - y\|^2, \mathrm{cost}(y, C)\right\} + \mathrm{cost}(y, C) \cdot \min\left\{\|x - y\|^2, \mathrm{cost}(x, C)\right\}.$$

Note that $f(x,y) = f(y,x)$. Then,

$$\mathbb{E}_c\left[\text{cost}(P,C')\right] = \frac{1}{2\text{cost}(P,C)} \sum_{(x,y)\in P\times P} f(x,y).$$

We now give an upper bound on $f(x,y)$ and then use this bound to finish the proof of Lemma 4.1.

Lemma 4.2. *For any $x, y \in P$, we have $f(x,y) \le 5\min\{\text{cost}(x,C), \text{cost}(y,C)\}\,\|x-y\|^2$.*

Proof. Since $f(x,y)$ is a symmetric function with respect to x and y, we may assume without loss of generality that $\text{cost}(x,C) \le \text{cost}(y,C)$. Then, we need to show that $f(x,y) \le 5\text{cost}(x,C)\|x-y\|^2$. Consider three cases.

Case 1: If $\text{cost}(x,C) \le \text{cost}(y,C) \le \|x-y\|^2$, then

$$f(x,y) = 2\text{cost}(x,C)\text{cost}(y,C) \le 2\text{cost}(x,C)\|x-y\|^2.$$

Case 2: If $\text{cost}(x,C) \le \|x-y\|^2 \le \text{cost}(y,C)$, then

$$f(x,y) = \text{cost}(x,C)\|x-y\|^2 + \text{cost}(y,C)\text{cost}(x,C).$$

By the triangle inequality, we have

$$\text{cost}(y,C) \le \left(\sqrt{\text{cost}(x,C)} + \|x-y\|\right)^2 \le 4\|x-y\|^2.$$

Thus, $f(x,y) \le 5\text{cost}(x,C)\|x-y\|^2$.

Case 3: If $\|x-y\|^2 \le \text{cost}(x,C) \le \text{cost}(y,C)$, then

$$f(x,y) = (\text{cost}(x,C) + \text{cost}(y,C))\,\|x-y\|^2.$$

By the triangle inequality,

$$\text{cost}(y,C) \le \left(\sqrt{\text{cost}(x,C)} + \|x-y\|\right)^2 \le 4\text{cost}(x,C).$$

Thus, we have $f(x,y) \le 5\text{cost}(x,C)\|x-y\|^2$.

In all cases, the desired inequality holds. This concludes the proof of Lemma 4.2. □

We use Lemma 4.2 to bound the expected cost of P. Let ϕ^* be a vector in $\mathbb{R}^P$ with $\phi^*_x = \text{cost}(x,C)$ for any $x \in P$. Then, $f(x,y) \le 5\min\left\{\phi^*_x, \phi^*_y\right\}\|x-y\|^2$. Since $\text{cost}(P,C) = \sum_{z\in P}\phi^*_z$, we have

$$\mathbb{E}_c\left[\text{cost}(P,C')\right] \le \underbrace{\frac{5\sum_{(x,y)\in P\times P}\min\left\{\phi^*_x,\phi^*_y\right\}\|x-y\|^2}{2\sum_{z\in P}\phi^*_z}}_{5F(\phi^*)}.$$

For arbitrary vector $\phi \in \mathbb{R}^P_{\ge 0}$, define the following function:

$$F(\phi) = \frac{\sum_{(x,y)\in P\times P}\min\{\phi_x,\phi_y\}\|x-y\|^2}{2\sum_{z\in P}\phi_z}. \tag{3}$$

We have $\mathbb{E}_c[\text{cost}(P, C')] \le 5F(\phi^*)$. Thus, to finish the proof of Lemma 4.1, it suffices to show that $F(\phi) \le \text{OPT}_1(P)$ for every $\phi \ge 0$ and particularly for $\phi = \phi^*$. By Lemma 4.3 (which we state and prove below), function $F(\phi)$ is maximized when $\phi \in \{0,1\}^P$. Let ϕ^{**} be a maximizer of $F(\phi)$ in $\{0,1\}^P$ and $P' = \{x \in P : \phi^{**}_x = 1\}$. Observe that

$$F(\phi^{**}) = \frac{\sum_{(x,y)\in P'\times P'} \|x-y\|^2}{2|P'|} = \text{OPT}_1(P').$$

Here we used the closed form expression (1) for the optimal cost of cluster P'. Since $P' \subset P$, we have $\text{OPT}_1(P') \le \text{OPT}_1(P)$. Thus, $F(\phi^*) \le F(\phi^{**}) \le \text{OPT}_1(P)$. □

Lemma 4.3. *There exists a maximizer ϕ^{**} of $F(\phi)$ in the region $\{\phi \ge 0\}$ such that $\phi \in \{0,1\}^P$.*

Proof. Let $m = |P|$ be the size of the cluster P and Π be the set of all bisections or permutations $\pi : \{1, \dots, m\} \to P$. Partition the set $\{\phi \ge 0\}$ into $m!$ regions ("cones over order polytopes"):

$$\{\phi : \phi \ge 0\} = \cup_{\pi\in\Pi} O_\pi,$$

where $O_\pi = \{\phi : 0 \le \phi_{\pi(1)} \le \phi_{\pi(2)} \le \cdots \le \phi_{\pi(m)}\}$. We show that for every $\pi \in \Pi$, there exists a maximizer ϕ^{**} of $F(\phi)$ in the region O_π, such that $\phi^{**} \in \{0,1\}^P$. Therefore, there exists a global maximizer ϕ^{**} that belongs $\{0,1\}^P$

Fix a $\pi \in \Pi$. Denote by V the hyperplane $\{\phi : \sum_{x\in P} \phi_x = 1\}$. Observe that F is a scale invariant function i.e., $F(\phi) = F(\lambda\phi)$ for every $\lambda > 0$. Thus, for every $\phi \in O_\pi$, there exists a $\phi' \in O_\pi \cap V$ (namely, $\phi' = \phi/(\sum_{x\in P}\phi_x)$) such that $F(\phi') = F(\phi)$. Hence, $\max\{F(\phi) : \phi \in O_\pi\} = \max\{F(\phi) : \phi \in O_\pi \cap V\}$. Note that for $\phi \in V$, the denominator of (3) equals 2, and for $\phi \in O_\pi$, the numerator of (3) is a linear function of ϕ. Therefore, $F(\phi)$ is a linear function in the convex set $O_\pi \cap V$. Consequently, one of the maximizers of F must be an extreme point of $O_\pi \cap V$.

The polytope $O_\pi \cap V$ is defined by m inequalities and one equality. Thus, for every extreme point ϕ of this polytope, all inequalities $\phi_{\pi(i)} \le \phi_{\pi(i+1)}$ but one must be tight. In other words, for some $j < m$, we have

$$0 = \phi_{\pi(1)} = \cdots = \phi_{\pi(j)} < \phi_{\pi(j+1)} = \cdots = \phi_{\pi(m)}. \tag{4}$$

Therefore, there exists a maximizer ϕ of $F(\phi)$ in $O_\pi \cap V$ satisfying (4) for some j. After rescaling ϕ – multiplying all coordinates of ϕ by $(m-j)$ – we obtain a vector ϕ^{**} whose first j coordinates $\phi^{**}_{\pi(1)}, \dots, \phi^{**}_{\pi(j)}$ are zeroes and the last $m-j$ coordinates $\phi^{**}_{\pi(j+1)}, \dots, \phi^{**}_{\pi(m)}$ are ones. Thus, $\phi^{**} \in \{0,1\}^P$. Since F is rescaling invariant, $F(\phi^{**}) = F(\phi)$. This concludes the proof. □

Replacing the bound in Lemma 3.2 from the analysis of Arthur and Vassilvitskii (2007) by our bound from Lemma 4.1 gives the following result (see also Lemma 5.6).

Theorem 4.4. *The approximation factor of k-means++ is at most* $5(\ln k + 2)$.

We now state an important corollary of Lemma 4.1.

Corollary 4.5. *For every $P \in \mathcal{P}$, the process $\widetilde{H}_t(P)$ for k-means++ is a supermartingale i.e.,*

$$\mathbb{E}\left[\widetilde{H}_{t+1}(\mathbf{X}) \mid C_t\right] \le \widetilde{H}_t(\mathbf{X}).$$

Proof. The value of $\widetilde{H}_t(\mathbf{X})$ changes only if at step t, we cover a yet uncovered cluster P. In this case, the value of $\widetilde{H}_{t+1}(P)$ changes by the new cost of P minus $5\mathrm{OPT}(P)$. By Lemma 4.1 this quantity is non-positive in expectation. □

Since the process $\widetilde{H}_t(P)$ is a supermartingale, we have $\mathbb{E}[\widetilde{H}_t(P)] \leq \widetilde{H}_0(P) = 5\mathrm{OPT}_1(P)$. Hence, $\mathbb{E}[H_t(P)] \leq \mathbb{E}[\widetilde{H}_t(P)] = 5\mathrm{OPT}_1(P)$. Thus, $\mathbb{E}[H_t(X)] \leq 5\mathrm{OPT}_k(\mathbf{X})$. Since $\mathrm{cost}_t(\mathbf{X}) = H_t(\mathbf{X}) + U_t(\mathbf{X})$ and we have a bound on the expectation of the covered cost, $H_t(\mathbf{X})$, in the remaining sections, we shall only analyze the uncovered cost $U_t(\mathbf{X})$.

5 Bi-criteria approximation of k-means++

In this section, we give a bi-criteria approximation guarantee for k-means++.

Theorem 5.1. *Let* $\mathrm{cost}_{k+\Delta}(\mathbf{X})$ *be the cost of the clustering with* $k+\Delta$ *centers sampled by the* k*-means++ algorithm. Then, for* $\Delta \geq 1$*, the expected cost* $\mathbb{E}[\mathrm{cost}_{k+\Delta}(\mathbf{X})]$ *is upper bounded by (below* $(a)^+$ *denotes* $\max(a,0)$*).*

$$\min\left\{2 + \frac{1}{2e} + \left(\ln\frac{2k}{\Delta}\right)^+, 1 + \frac{k}{e\,(\Delta-1)}\right\} 5\mathrm{OPT}_k(\mathbf{X}).$$

Note that the above approximation guarantee is the minimum of two bounds: (1) $2 + \frac{1}{2e} + \ln\frac{2k}{\Delta}$ for $1 \leq \Delta \leq 2k$; and (2) $1 + \frac{k}{e(\Delta-1)}$ for $\Delta \geq 1$. The second bound is stronger than the first bound when $\Delta/k \gtrapprox 0.085$.

5.1 Proof overview of Theorem 5.1

We now present a high level overview of the proof and then give a formal proof. Our proof consists of three steps.

First, we prove bound (2) on the expected cost of the clustering returned by k-means++ after $k+\Delta$ rounds. We argue that the expected cost of the covered clusters is bounded by $5\mathrm{OPT}_k(\mathbf{X})$ (see Section 3) and thus it is sufficient to bound the expected cost of uncovered clusters. Consider an optimal cluster $P \in \mathcal{P}$. We need to estimate the probability that it is not covered after $k+\Delta$ rounds. We upper bound this probability by the probability that the algorithm does not cover P before it makes Δ misses (note: after $k+\Delta$ rounds k-means++ must make at least Δ misses).

In this overview, we make the following simplifying assumptions (which turn out to be satisfied in the worst case for bi-criteria k-means++): Suppose that the uncovered cost of cluster P does not decrease before it is covered and equals $U(P)$ and, moreover, the total cost of all covered clusters almost does not change and equals $H(\mathbf{X})$ (this may be the case if one large cluster contributes most of the covered cost, and that cluster is covered at the first step of k-means++). Under these assumptions, the probability that k-means++ chooses Δ centers in the already covered clusters and does not choose a single center in P equals $(H(\mathbf{X})/(U(P)+H(\mathbf{X})))^\Delta$. If k-means++ does not choose a center in P, the *uncovered* cost of cluster P is $U(P)$; otherwise, the *uncovered* cost of cluster P is 0. Thus, the expected *uncovered cost* of P is $(H(\mathbf{X})/(U(P)+H(\mathbf{X})))^\Delta U(P)$. It is easy to show that $(H(\mathbf{X})/(U(P)+H(\mathbf{X})))^\Delta U(P) \leq H(\mathbf{X})/(e(\Delta-1))$. Thus, the expected *uncovered cost* of all clusters is at most

$$\frac{k}{(e(\Delta-1))}\mathbb{E}[H(\mathbf{X})] \leq \frac{k}{(e(\Delta-1))} 5\mathrm{OPT}_k(\mathbf{X}).$$

Then, we use ideas from Arthur and Vassilvitskii (2007), Dasgupta (2013) to prove the following statement: Let us count the cost of uncovered clusters only when the number of misses after k rounds of k-means++ is greater than $\Delta/2$. Then the expected cost of uncovered clusters is at most $O(\log(k/\Delta)) \cdot \mathrm{OPT}_k(\mathbf{X})$. That is, $\mathbb{E}[H(U_k(\mathbf{X}) \cdot \mathbf{1}\{M(C_k) \geq \Delta/2\}] \leq O(\log(k/\Delta)) \cdot \mathrm{OPT}_k(\mathbf{X})$.

Finally, we combine the previous two steps to get bound (1). We argue that if the number of misses after k rounds of k-means++ is less than $\Delta/2$, then almost all clusters are covered. Hence, we can apply bound (2) to $k' \leq \Delta/2$ uncovered clusters and Δ remaining rounds of k-means++ and get a $5(1+1/(2e))$ approximation. If the number of misses is greater than $\Delta/2$, then the result from the previous step yields an $O(\log(k/\Delta))$ approximation.

5.2 Analysis of k-means++

In this section, we analyze the bi-criteria k-means++ algorithm and prove Theorem 5.1. To this end, we establish the first and second bounds from Theorem 5.1 on the expected cost of the clustering after $k + \Delta$ rounds of k-means. We will start with the second bound.

5.2.1 Bi-criteria bound for large Δ

Lemma 5.2. *The following bi-criteria bound holds*

$$\mathbb{E}\left[\mathrm{cost}_{k+\Delta}\left(\mathbf{X}\right)\right] \leq 5\left(1 + \frac{k}{e\left(\Delta - 1\right)}\right)\mathrm{OPT}_k(\mathbf{X}).$$

Consider the discrete time Markov chain C_t associated with k-means++ algorithm (see Section 3). Let $P \in \mathcal{P}$ be an arbitrary cluster in the optimal solution. Partition all states of the Markov chain into $k + \Delta$ disjoint groups $\mathcal{M}_0, \mathcal{M}_1, \cdots, \mathcal{M}_{k+\Delta-1}$ and $\mathcal{H}$. Each set $\mathcal{M}_i$ contains all states C with i misses that do not cover P: $\mathcal{M}_i = \{C : M(C) = i, P \cap C = \varnothing\}$. The set $\mathcal{H}$ contains all states C that cover P: $\mathcal{H} = \{C : P \cap C \neq \varnothing\}$.

We now define a new Markov chain X_t. To this end, we first expand the set of states $\{C\}$. For every state C of the process C_t, we create two additional "virtual" states C^a and C^b. Then, we let $X_{2t} = C_t$ for every even step $2t$, and

$$X_{2t+1} = \begin{cases} C_t^a, & \text{if } C_{t+1} \in \mathcal{M}_i \\ C_t^b, & \text{if } C_{t+1} \in \mathcal{M}_{i+1} \cup \mathcal{H}. \end{cases}$$

for every odd step $2t + 1$. We stop X_t when C_t stops or when C_t hits the set $\mathcal{H}$ (i.e., $C_t \in \mathcal{H}$). Loosely speaking, X_t follows Markov chain C_t but makes additional intermediate stops. When C_t moves from one state in $\mathcal{M}_i$ to another state in $\mathcal{M}_i$, X_{2t+1} stops in C_t^a; and when C_t moves from a state in $\mathcal{M}_i$ to a state in $\mathcal{M}_{i+1}$ or $\mathcal{H}$, X_{2t+1} stops in C_t^b.

Write transition probabilities for X_t:

$$\mathbb{P}\left[X_{2t+1} = C^a \mid X_{2t} = C\right] = \frac{U(\mathbf{X}, C) - U(P, C)}{\mathrm{cost}(\mathbf{X}, C)},$$

$$\mathbb{P}\left[X_{2t+1} = C^b \mid X_{2t} = C\right] = \frac{U(P, C) + H(\mathbf{X}, C)}{\mathrm{cost}(\mathbf{X}, C)},$$

and for all $C \in \mathcal{M}_i$ and $C' = C \cup \{x\} \in \mathcal{M}_i$,

$$\mathbb{P}\left[X_{2t+2} = C' \mid X_{2t+1} = C^a\right] = \frac{\mathrm{cost}(x, C)}{U(\mathbf{X}, C) - U(P, C)},$$

for all $C \in \mathcal{M}_i$ and $C' = C \cup \{x\} \in \mathcal{M}_{i+1} \cup \mathcal{H}$,

$$\mathbb{P}\left[X_{2t+2} = C' \mid X_{2t+1} = C^b\right] = \frac{\text{cost}(x, C)}{U(P, C) + H(\mathbf{X}, C)}.$$

Above, $U(\mathbf{X}, C) - U(P, C)$ is the cost of points in all uncovered clusters except for P. If we pick a center from these clusters, we will necessarily cover a new cluster, and therefore X_{2t+2} will stay in $\mathcal{M}_i$. Similarly, $U(P, C) + H(\mathbf{X}, C)$ is the cost of all covered clusters plus the cost of P. If we pick a center from these clusters, then X_{2t+2} will move to $\mathcal{M}_{i+1}$ or $\mathcal{H}$.

Define another Markov chain $\{Y_t\}$. The transition probabilities of $\{Y_t\}$ are the same as the transition probabilities of X_t except Y never visits states in $\mathcal{H}$ and therefore for $C \in \mathcal{M}_i$ and $C' = C \cup \{x\} \in \mathcal{M}_{i+1}$, we have

$$\mathbb{P}\left[Y_{2t+2} = C' \mid Y_{2t+1} = C^b\right] = \frac{\text{cost}(x, C)}{H(\mathbf{X}, C)}.$$

We now prove a lemma that relates probabilities of visiting states by X_t and Y_t.

Lemma 5.3. *For every $t \le k + \Delta$ and states $C' \in \mathcal{M}_i$, $C'' \in \mathcal{M}_\Delta$, we have*

$$\frac{\mathbb{P}\left[C'' \in \{X_j\} \mid X_{2t} = C'\right]}{\mathbb{P}\left[C'' \in \{Y_j\} \mid Y_{2t} = C'\right]} \le \left(\frac{\widetilde{H}(\mathbf{X}, C'')}{\widetilde{H}(\mathbf{X}, C'') + U(P, C'')}\right)^{\Delta - i}$$

where $\{C'' \in \{X_j\}\}$ and $\{C'' \in \{Y_j\}\}$ denote the events X visits C'' and Y visits C'', respectively.

Proof. Consider the unique path p from C' to C'' in the state space of X (note that the transition graphs for X and Y are directed trees). The probability of transitioning from C' to C'' for X and Y equals the product of respective transition probabilities for every edge on the path. Recall that transitions probabilities for X and Y are the same for all states but C^b, where $C \in \cup_j \mathcal{M}_j$. The number of such states on the path p is equal to the number transitions from $\mathcal{M}_j$ to $\mathcal{M}_{j+1}$, since X and Y can get from $\mathcal{M}_j$ to $\mathcal{M}_{j+1}$ only through a state C^b on the boundary of $\mathcal{M}_j$ and $\mathcal{M}_{j+1}$. The number of transitions from $\mathcal{M}_j$ to $\mathcal{M}_{j+1}$ equals $\Delta - i$. For each state C^b on the path, the ratio of transition probabilities from C^b to the next state $C \cup \{x\}$ for Markov chains X and Y equals

$$\frac{H(\mathbf{X}, C)}{U(P, C) + H(\mathbf{X}, C)} \le \frac{\widetilde{H}(\mathbf{X}, C'')}{U(P, C'') + \widetilde{H}(\mathbf{X}, C'')},$$

here we used that (a) $U(P, C) \ge U(P, C'')$ since $U_t(P)$ is a non-increasing process; and (b) $H(P, C) \le \widetilde{H}(P, C'')$ since $H_t(P) \le \widetilde{H}_{t'}(P)$ if $t \le t'$ (see Section 3). ☐

We now prove an analog of Corollary 4.5 for $\widetilde{H}(\mathbf{X}, Y_j)$.

Lemma 5.4. *$\widetilde{H}(\mathbf{X}, Y_t)$ is a supermartingale.*

Proof. If $Y_j = C$, then Y_{j+1} can only be in $\left\{C^a, C^b\right\}$. Since $\widetilde{H}(\mathbf{X}, C^a) = \widetilde{H}(\mathbf{X}, C^b) = \widetilde{H}(\mathbf{X}, C)$, we have $\mathbb{E}\left[\widetilde{H}(\mathbf{X}, Y_{j+1}) \mid Y_j = C\right] = \widetilde{H}(\mathbf{X}, Y_j)$.

If $Y_j = C^a$, then $Y_{j+1} = C'$ where the new center c should be in uncovered clusters with respect to C_t.

$$\mathbb{E}\left[H(P', Y_{j+1}) \mid Y_j = C^a, c \in P'\right] \le 5\text{OPT}_1(P'),$$

which implies

$$\mathbb{E}\left[\widetilde{H}(P', Y_{j+1}) \mid Y_j = C^a, c \in P'\right] \leq \widetilde{H}(P', Y_j).$$

Therefore, we have

$$\mathbb{E}\left[\widetilde{H}(\mathbf{X}, Y_{j+1}) \mid Y_j = C^a\right] \leq \widetilde{H}(\mathbf{X}, Y_j).$$

If $Y_j = C^b$, then for any possible state C' of Y_{j+1}, the new center should be in covered clusters with respect to C. By definition, we must have $\widetilde{H}(\mathbf{X}, C') = \widetilde{H}(\mathbf{X}, C) = \widetilde{H}(\mathbf{X}, C^b)$. Thus, it holds that $\mathbb{E}\left[\widetilde{H}(\mathbf{X}, Y_{j+1}) \mid Y_j = C^b\right] = \widetilde{H}(\mathbf{X}, Y_j)$.

Combining all these cases, we get $\left\{\widetilde{H}(\mathbf{X}, Y_j)\right\}$ is a supermartingale. □

We now use Lemma 5.3 and Lemma 5.4 to bound the expected uncovered cost of P after $k + \Delta$ rounds of k-means++.

Lemma 5.5. *For any cluster $P \in \mathcal{P}$ and $t \leq k + \Delta$, we have*

$$\mathbb{E}\left[U_{k+\Delta}(P) \mid C_t\right] \leq \frac{\widetilde{H}_t(\mathbf{X})}{e(\Delta - M(C_t) - 1)}.$$

Proof. Since k-means++ samples $k + \Delta$ centers and the total number of clusters in the optimal solution $\mathcal{P}$ is k, k-means++ must make Δ misses. Hence, the process $\{X_t\}$ which follows k-means++ must either visit a state in $\mathcal{M}_{\geq\Delta}$ or stop in $\mathcal{H}$ (recall that we stop process X_t if it reaches $\mathcal{H}$).

If $\{X_t\}$ stops in group $\mathcal{H}$, then the cluster P is covered which means that $U_{k+\Delta}(P) = 0$. Let $\partial\mathcal{M}_\Delta$ be the frontier of $\mathcal{M}_\Delta$ i.e., the states that X_t visits first when it reaches $\mathcal{M}_\Delta$ (recall that the transition graph of X_t is a tree). The expected cost $\mathbb{E}\left[U_{k+\Delta}(P) \mid C_t\right]$ is upper bounded by the expected uncovered cost of P at time when C_t reaches $\partial\mathcal{M}_\Delta$. Thus,

$$\mathbb{E}\left[U_{k+\Delta}(P) \mid C_t\right] \leq \sum_{C \in \partial\mathcal{M}_\Delta} \mathbb{P}\left[C \in \{X_j\} \mid C_t\right] U(P, C).$$

Observe that by Lemma 5.3, for any $C \in \partial\mathcal{M}_\Delta$, we have

$$\mathbb{P}\left[C \in \{X_j\} \mid C_t\right] U(P, C) \leq \mathbb{P}\left[C \in \{Y_j\} \mid C_t\right] \left(\frac{\widetilde{H}(\mathbf{X}, C)}{\widetilde{H}(\mathbf{X}, C) + U(P, C)}\right)^{\Delta'} U(P, C).$$

Let $f(x) = x(1/(1+x))^{\Delta'}$. Then, $f(x)$ is maximized at $x = 1/(\Delta' - 1)$ and the maximum value $f(1/(\Delta' - 1)) = 1/(e(\Delta' - 1))$. Therefore, for every $C \in \partial\mathcal{M}_\Delta$, we have

$$\begin{aligned}\mathbb{P}[C \in \{X_j\} \mid C_t] U(P, C) &\leq \mathbb{P}\left[C \in \{Y_j\} \mid C_t\right] f\left(\frac{U(P, C)}{\widetilde{H}(\mathbf{X}, C)}\right) \widetilde{H}(\mathbf{X}, C) \\ &\leq \mathbb{P}\left[C \in \{Y_j\} \mid C_t\right] \frac{\widetilde{H}(\mathbf{X}, C)}{e(\Delta' - 1)}.\end{aligned}$$

Let $\tau = \min\{j : Y_j \in \partial\mathcal{M}_\Delta\}$ be the stopping time when Y_j first visits $\partial\mathcal{M}_\Delta$. We get

$$\sum_{C \in \partial\mathcal{M}_\Delta} \mathbb{P}\left[C \in \{Y_j\} \mid C_t\right] \widetilde{H}(\mathbf{X}, C) = \mathbb{E}\left[\widetilde{H}(\mathbf{X}, Y_\tau) \mid C_t\right].$$

By Lemma 5.4, $\widetilde{H}(\mathbf{X}, Y_j)$ is a supermartingale. Thus, by the optional stopping theorem,

$$\mathbb{E}\left[\widetilde{H}(\mathbf{X}, Y_\tau) \mid C_t\right] \leq \widetilde{H}(\mathbf{X}, C_t).$$

Therefore, we have

$$\mathbb{E}\left[U_{k+\Delta}(P) \mid C_t\right] \leq \frac{\widetilde{H}_t(\mathbf{X})}{e(\Delta - M(C_t) - 1)},$$

This concludes the proof. □

We now add up bounds from Lemma 5.5 with $t = 0$ for all clusters $P \in \mathcal{P}$ and obtain Lemma 5.2.

5.3 Bi-criteria bound for small Δ

In this section, we give another bi-criteria approximation guarantee for k-means++.

Lemma 5.6. *Let* $\mathrm{cost}_{k+\Delta}(\mathbf{X})$ *be the cost of the the clustering resulting from sampling* $k+\Delta$ *centers according to the* k*-means++ algorithm (for* $\Delta \in \{1, \ldots, 2k\}$*). Then,*

$$\mathbb{E}\left[\mathrm{cost}_{k+\Delta}(X)\right] \leq 5\left(2 + \frac{1}{2e} + \ln\frac{2k}{\Delta}\right)\mathrm{OPT}_k(X).$$

Proof. Consider k-means++ clustering algorithm and the corresponding random process C_t. Fix a $\kappa \in \{1, \ldots, k\}$. Let τ be the first iteration[1] (stopping time) when $K(C_\tau) \leq \kappa$ if $K(C_k) \leq \kappa$; and $\tau = k$, otherwise. We refer the reader to Section 3 for definitions of $M(C_t)$, $U_t(X) = U(X, C_t)$, and $K(C_t)$.

We separately analyze the cost of uncovered clusters after the first τ steps and the last $k' - \tau$ steps, where $k' = k + \Delta$ is the total number of centers chosen by k-means++.

The first step of our proof follows the analysis of k-means++ by Dasgupta (2013), and by Arthur and Vassilvitskii (2007). Define a potential function Ψ (see Dasgupta 2013):

$$\Psi_t := \frac{M(C_t)U(X, C_t)}{K(C_t)}.$$

If $K(C_t) = 0$, then $M(C_t)$ and $U(X, C_t)$ must be 0 and we let $\Psi_t = 0$

We use the following result by Dasgupta (2013) to estimate $\mathbb{E}[\Psi_\tau(X)]$ in Lemma 5.8.

Lemma 5.7 (Dasgupta (2013))**.** *For any* $0 \leq t \leq k$*, we have*

$$\mathbb{E}\left[\Psi_{t+1} - \Psi_t \mid C_t\right] \leq \frac{H(X, C_t)}{K(C_t)}.$$

Lemma 5.8. *Then, the following bound holds:*

$$\mathbb{E}[\Psi_\tau(X)] \leq 5\left(1 + \ln\left(\frac{k}{\kappa + 1}\right)\right)\mathrm{OPT}_k(X).$$

[1] Recall, that $K(C_t)$ is a non-increasing stochastic process with $K(C_0) = k$.

Proof. Note that $\Psi_1 = 0$ as $M(C_1) = 0$. Thus,

$$\mathbb{E}[\Psi_\tau] \leq \sum_{t=1}^{\tau-1} \mathbb{E}\big[\Psi_{t+1} - \Psi_t\big] \leq \mathbb{E}\Big[\sum_{t=1}^{\tau-1} \frac{H(X, C_t)}{K(C_t)}\Big].$$

Using the inequality $H(X, C_t) \leq \widetilde{H}_k(X)$ (see Section 3), we get:

$$\mathbb{E}[\Psi_\tau] \leq \mathbb{E}\Big[\sum_{t=1}^{\tau-1} \frac{\widetilde{H}_k(X)}{K(C_t)}\Big] \leq \mathbb{E}\Big[\widetilde{H}_k(X) \cdot \sum_{t=1}^{\tau-1} \frac{1}{K(C_t)}\Big].$$

Observe that $K(C_1), \ldots, K(C_{\tau-1})$ is a non-increasing sequence in which two consecutive terms are either equal or $K(C_{i+1}) = K(C_i) - 1$. Moreover, $K(C_1) = k$ and $K(C_{\tau-1}) > \kappa$. Therefore, by Lemma 5.9 (see below), for every realization $C_0, C_1, \ldots, C_\tau$, we have:

$$\sum_{t=1}^{\tau-1} \frac{1}{K(C_t)} \leq 1 + \log {}^{k}\!/_{(\kappa+1)}.$$

Thus,

$$\mathbb{E}[\Psi_\tau] \leq (1 + \log {}^{k}\!/_{(\kappa+1)})\mathbb{E}[\widetilde{H}_k(X)] \leq 5(1 + \log {}^{k}\!/_{(\kappa+1)})\ \mathrm{OPT}_k(X).$$

This concludes the proof. □

Let $\kappa = \lfloor(\Delta - 1)/2\rfloor$. By Lemma 5.8, we have

$$\mathbb{E}\Big[\frac{M(C_\tau)U_\tau(X)}{K(C_\tau)}\Big] \leq 5\left(1 + \ln\frac{2k}{\Delta}\right)\mathrm{OPT}_k(X).$$

Since $U_t(X)$ is a non-increasing stochastic process, we have $\mathbb{E}[U_{k+\Delta}(X)] \leq \mathbb{E}[U_\tau(X)]$. Thus,

$$\mathbb{E}\Big[\frac{M(C_\tau)}{K(C_\tau)} \cdot U_{k+\Delta}(X)\Big] \leq 5\left(1 + \ln\frac{2k}{\Delta}\right)\mathrm{OPT}_k(X).$$

Our goal is to bound $\mathbb{E}[U_{k'}(X)]$. Write,

$$\mathbb{E}[U_{k'}(X)] = \mathbb{E}\Big[\frac{M(C_\tau)}{K(C_\tau)} \cdot U_{k'}(X)\Big] + \mathbb{E}\Big[\frac{K(C_\tau) - M(C_\tau)}{K(C_\tau)} \cdot U_{k'}(X)\Big].$$

The first term on the right hand side is upper bounded by $5(1 + \ln\frac{2k}{\Delta})\mathrm{OPT}_k(X)$. We now estimate the second term, which we denote by $(*)$.

Note that $K(C_t) - M(C_t) = k - t$, since the number of uncovered clusters after t steps of k-means++ equals the number of misses plus the number of steps remaining. Particularly, if $\tau = k$, we have $K(C_\tau) - M(C_\tau) = K(C_k) - M(C_k) = 0$. Consequently, if $\tau = k$, then the second term $(*)$ equals 0. Thus, we only need to consider the case, when $\tau < k$. Note that in this case $K(C_\tau) = \kappa$. By Lemma 5.2 (applied to all uncovered clusters), we have

$$\mathbb{E}[U_{k'}(X) \mid C_\tau, \tau] \leq \frac{K(C_\tau)}{e(\Delta' - 1)}\widetilde{H}_\tau(X),$$

where $\Delta' = \Delta - M(C_\tau)$.

Thus,

$$\mathbb{E}\Big[\frac{K(C_\tau)-M(C_\tau)}{K(C_\tau)}\cdot U_{k'}(X)\mid C_\tau,\tau\Big]\le\frac{K(C_\tau)-M(C_\tau)}{K(C_\tau)}\cdot\frac{K(C_\tau)}{e(\Delta'-1)}\cdot\widetilde{H}_\tau(X)=(**).$$

Plugging in $K(C_\tau)=\kappa$ and the expression for Δ' (see above), and using that $\kappa\le(\Delta-1)/2$, we get

$$(**)=\frac{\kappa-M(C_\tau)}{e(\Delta-M(C_\tau)-1)}.\cdot\widetilde{H}_\tau(X)\le\frac{1}{2e}\widetilde{H}_\tau(X).$$

Finally, taking the expectation over all C_τ, we obtain the bound

$$\mathbb{E}\Big[\frac{K(C_\tau)-M(C_\tau)}{K(C_\tau)}\cdot U_{k'}(X)\Big]\le\frac{5\mathrm{OPT}_1(X)}{2e}.$$

Thus, $\mathbb{E}[U_{k'}(X)]\le 5(1+{}^1\!/_{2e}+\ln{}^{2k}\!/_{\Delta})\mathrm{OPT}_k(X)$. Therefore,

$$\mathbb{E}[\mathrm{cost}_{k'}(X)]=\mathbb{E}[H_{k'}(X)]+U_{k'}(X)\le 5(2+\frac{1}{2e}+\ln\frac{2k}{\Delta})\ \mathrm{OPT}_k(X).$$

□

We now prove Lemma 5.9.

Lemma 5.9. *For any $t\le k$ integers $a_1\ge a_2\ge\cdots\ge a_t$ such that $a_1=k$, $a_t>\kappa$ and $a_i-a_{i+1}\in\{0,1\}$ for all $1\le i<t$, the following inequality holds*

$$\sum_{i=1}^{t}\frac{1}{a_i}\le 1+\log\left(\frac{k}{\kappa+1}\right).$$

Proof. It is easy to see that the sum is maximized when $t=k$, and the sequence $a_1,\ldots,a_k$ is as follows:

$$\underbrace{\frac{1}{k},\frac{1}{k-1},\ldots,\frac{1}{\kappa+2}}_{(k-(\kappa+1))\text{ terms}},\underbrace{\frac{1}{\kappa+1},\ldots,\frac{1}{\kappa+1}}_{(\kappa+1)\text{ terms}}.$$

The sum of the first $(k-(\kappa+1))$ terms is upper bounded by

$$\int_{1/(\kappa+1)}^{1/k}\frac{1}{x}\,dx=\ln\frac{k}{\kappa+1}.$$

The sum of the last $(\kappa+1)$ terms is 1. □

6 Analysis of k-means‖

In this section, we give a sketch of analysis for the k-means‖ algorithm. Specifically, we show upper bounds on the expected cost of the solution after T rounds.

Theorem 6.1. *The expected cost of the clustering returned by k-means‖ algorithm after T rounds are upper bounded as follows:*

$$\text{for } \ell < k, \qquad \mathbb{E}\left[\text{cost}_{T+1}(\mathbf{X})\right] \le \left(e^{-\frac{\ell}{k}}\right)^T \mathbb{E}\left[\text{cost}_1(\mathbf{X})\right] + \frac{5\text{OPT}_k(\mathbf{X})}{1-e^{-\frac{\ell}{k}}};$$

$$\text{for } \ell \ge k, \qquad \mathbb{E}\left[\text{cost}_{T+1}(\mathbf{X})\right] \le \left(\frac{k}{e\ell}\right)^T \mathbb{E}\left[\text{cost}_1(\mathbf{X})\right] + \frac{5\text{OPT}_k(\mathbf{X})}{1-k/e\ell}.$$

Remark: For the second bound ($\ell \ge k$), the additive term $5\text{OPT}_k(\mathbf{X})/(1-k/(e\ell)) \le 8\text{OPT}_k(\mathbf{X})$.

The probability that a point is sampled by k-means‖ is strictly greater than the probability that it is sampled by k-means‖$_{\text{Pois}}$ since $1-e^{-\lambda} < \lambda$ for all $\lambda > 0$. Thus, for every round, we can couple k-means‖$_{\text{Pois}}$ and k-means‖ so that each point sampled by k-means‖$_{\text{Pois}}$ is also sampled by k-means‖. Thus, the expected cost returned by k-means‖ is at most the expected cost returned by k-means‖$_{\text{Pois}}$. In the following analysis, we show an upper bound for the expected cost of the solution returned by k-means‖$_{\text{Pois}}$.

As a thought experiment, consider a modified k-means‖$_{\text{Pois}}$ algorithm. This algorithm is given the set $\mathbf{X}$, parameter k, and additionally the optimal solution $\mathcal{P} = \{P_1, \dots, P_k\}$. Although this modified algorithm is useless in practice as we do not know the optimal solution in advance, it will be helpful for our analysis.

In every round t, the modified algorithm first draws independent Poisson random variables $Z_t(P_i) \sim \text{Pois}(\lambda_t(P_i))$ for every cluster $i \in \{1, \dots, k\}$ with rate $\lambda_t(P_i) = \sum_{x \in P_i} \lambda_t(x)$. Then, for each $i \in \{1, \dots, k\}$, it samples $Z_t(P_i)$ points $x \in P_i$ with repetitions from P_i, picking every point x with probability $\lambda_t(x)/\lambda_t(P_i)$ and adds them to the set of centers C_t. We assume that points in every set C_t are ordered in the same way as they were chosen by this algorithm.

We claim that the distribution of the output sets C_T of this algorithm is exactly the same as in the original k-means‖$_{\text{Pois}}$ algorithm. Therefore, we can analyze the modified algorithm instead of k-means‖$_{\text{Pois}}$, using the framework described in Sections 3.

Lemma 6.2. *The sets C_t in the original and modified k-means‖$_{\text{Pois}}$ algorithms are identically distributed.*

Proof. Consider $|P_i|$ independent Poisson point processes $N_x(a)$ with rates $\lambda_t(x)$, where $x \in P_i$ (here, we use variable a for time). Suppose we add a center x at step t of the algorithm if $N_x(t) \ge 1$. On the one hand, the probability that we choose x is equal to $1 - e^{-\lambda_t(x)}$ which is exactly the probability that k-means‖$_{\text{Pois}}$ picks x as a center at step t. On the other hand, the sum $N_{P_i} = \sum_{x \in P_i} N_x$ is a Poisson point process with rate $\lambda_t(P_i)$. Thus, the total number of jumps in the interval $[0, 1]$ of processes N_x with $x \in P_i$ is distributed as $Z_t(P_i)$. Moreover, the probability that N_x jumps at time a conditioned on the event that N_{P_i} jumps at time a is $\lambda_t(x)/\lambda_t(P_i)$. Thus, for every jump of N_{P_i}, we choose one random center x with probability $\lambda_t(x)/\lambda_t(P_i)$. □

Lemma 6.3. *For k-means‖ algorithm with parameter ℓ, the following bounds hold:*

$$\text{for } \ell < k, \qquad \mathbb{E}\left[\text{cost}_{t+1}(\mathbf{X})\right] \le e^{-\frac{\ell}{k}} \cdot \mathbb{E}\left[\text{cost}_t(\mathbf{X})\right] + 5\text{OPT}_k(\mathbf{X});$$

$$\text{for } \ell \ge k, \qquad \mathbb{E}\left[\text{cost}_{t+1}(\mathbf{X})\right] \le \left(\frac{k}{e\ell}\right) \cdot \mathbb{E}\left[\text{cost}_t(\mathbf{X})\right] + 5\text{OPT}_k(\mathbf{X}).$$

Proof. Since the expected cost returned by k-means‖ is at most the expected cost returned by k-means‖$_{\text{Pois}}$, we analyze the expected cost of the clustering after one step of k-means‖$_{\text{Pois}}$.

If the algorithm covers cluster P_i at round t, then at the next round, its uncovered cost equals 0. The number of centers chosen in P_i is determined by the Poisson random variable $Z_{t+1}(P_i)$. Hence, P_i is uncovered at round $t+1$ only if $Z_{t+1}(P_i) = 0$. Since $U_t(P_i)$ is non-increasing in t and $U_t(P_i) \le \mathrm{cost}_t(P_i)$, we have

$$\mathbb{E}\left[U_{t+1}(P_i) \mid C_t\right] \le \mathbb{P}\left[Z_{t+1}(P_i) = 0\right] U_t(P_i) \le \exp\left(-\frac{\ell\,\mathrm{cost}_t(P_i)}{\mathrm{cost}_t(\mathbf{X})}\right)\mathrm{cost}_t(P_i).$$

Define two function: $f(x) = e^{-x}\cdot x$; and $g(x) = f(x)$ for $x \in [0,1]$ and $g(x) = e^{-1}$ for $x \in [1,\infty)$. Then,

$$\mathbb{E}\left[U_{t+1}(\mathbf{X}) \mid C_t\right] \le \left(\frac{1}{k}\sum_{i=1}^{k} f\left(\frac{\ell \mathrm{cost}_t(P_i)}{\mathrm{cost}_t(\mathbf{X})}\right)\right)\frac{k\mathrm{cost}_t(\mathbf{X})}{\ell}.$$

Since $g(x) \le f(x)$, and $g(x)$ is concave for $x \ge 0$, we have

$$\mathbb{E}\left[U_{t+1}(\mathbf{X}) \mid C_t\right] \le \left(\frac{1}{k}\sum_{i=1}^{k} g\left(\frac{\ell\ \mathrm{cost}_t(P_i)}{\mathrm{cost}_t(\mathbf{X})}\right)\right)\frac{k\mathrm{cost}_t(\mathbf{X})}{\ell} \le g\left(\frac{\ell}{k}\right)\frac{k\mathrm{cost}_t(\mathbf{X})}{\ell}.$$

Here, we use that $\sum_i \mathrm{cost}_t(P_i) = \mathrm{cost}_t(\mathbf{X})$.

Therefore, for $\ell \le k$, we have

$$\mathbb{E}\left[U_{t+1}(\mathbf{X}) \mid C_t\right] \le \left(e^{-\frac{\ell}{k}}\right)\mathrm{cost}_t(\mathbf{X});$$

and for $\ell \ge k$, we have

$$\mathbb{E}\left[U_{t+1}(\mathbf{X}) \mid C_t\right] \le \left(\frac{k}{e\ell}\right)\mathrm{cost}_t(\mathbf{X}).$$

Similar to Corollary 4.5, the process $\widetilde{H}_t(P)$ for k-means$\|_{\mathrm{Pois}}$ is also a supermartingale, which implies $\mathbb{E}\left[H_{t+1}(\mathbf{X})\right] \le 5\mathrm{OPT}_k(\mathbf{X})$. This concludes the proof. □

Proof of Theorem 6.1. Applying the bound from Lemma 6.3 for t times, we get the following results. For $\ell \le k$,

$$\mathbb{E}\left[\mathrm{cost}_{t+1}(\mathbf{X})\right] \le \left(e^{-\frac{\ell}{k}}\right)^t \mathbb{E}\left[\mathrm{cost}_1(\mathbf{X})\right] + 5\mathrm{OPT}_k(\mathbf{X})\eta_t,$$

where $\eta_t = \sum_{j=1}^{t}\left(e^{-\frac{\ell}{k}}\right)^{j-1} < \frac{1}{1-e^{-\frac{\ell}{k}}}$. For $\ell \ge k$,

$$\mathbb{E}\left[\mathrm{cost}_{t+1}(\mathbf{X})\right] \le \left(\frac{k}{e\ell}\right)^t \mathbb{E}\left[\mathrm{cost}_1(\mathbf{X})\right] + 5\mathrm{OPT}_k(\mathbf{X})\eta_t,$$

where $\eta_t = \sum_{j=1}^{t}\left(\frac{k}{e\ell}\right)^{j-1} \le \frac{1}{1-\frac{k}{e\ell}}$. □

Corollary 6.4. *Consider a data set* $\mathbf{X}$ *with more than* k *distinct points. Let*

$$T = \ln \mathbb{E}\left[\frac{\mathrm{cost}_1(\mathbf{X})}{\mathrm{OPT}_k(\mathbf{X}))}\right]$$

and $\ell > k$*. Then, after* T *rounds of k-means||, the expected cost of clustering* $\mathbb{E}\left[\mathrm{cost}_T(\mathbf{X})\right]$ *is at most* $9\mathrm{OPT}_k(\mathbf{X})$.

7 Exponential Race k-means++ and Reservoir Sampling

In this section, we show how to implement k-means++ algorithm in parallel using R passes over the data set. This implementation, which we refer to as k-means++$_{\mathrm{ER}}$ (exponential race k-means++), is very similar to k-means$\|$, but has stronger theoretical guarantees. Like k-means$\|$, in every round, k-means++$_{\mathrm{ER}}$ tentatively selects ℓ centers, in expectation. However, in the same round, it removes some of the just selected centers (without making another pass over the data set). Consequently, by the end of each iteration, the algorithm keeps at most k centers.

We can run k-means++$_{\mathrm{ER}}$ till it samples exactly k centers; in which case, the distribution of k sampled centers is identical to the distribution of the regular k-means++, and the expected number of rounds or passes over the data set R is upper bounded by

$$O\Big(\frac{k}{\ell} + \log \frac{\mathrm{OPT}_1(\mathbf{X})}{\mathrm{OPT}_k(\mathbf{X})}\Big).$$

We note that R is never greater than k. We can also run this algorithm for at most R^* rounds. Then, the expected cost of the clustering is at most

$$5(\ln k + 2)\,\mathrm{OPT}_k(\mathbf{X}) + 5R^*\left(\frac{4k}{e\ell R^*}\right)^{R^*} \cdot \mathrm{OPT}_1(\mathbf{X}).$$

7.1 Algorithm

In this section, we give a high level description of our k-means++$_{\mathrm{ER}}$ algorithm. In Section 7.2, we show how to efficiently implement k-means++$_{\mathrm{ER}}$ using lazy updates and explain why our algorithm makes R passes over the data set.

The algorithm simulates n continuous-time stochastic processes. Each stochastic process is associated with one of the points in the data set. We denote the process corresponding to $x \in \mathbf{X}$ by $P_t(x)$. Stochastic process $P_t(x)$ is a Poisson process with variable arrival rate $\lambda_t(x)$.

The algorithm chooses the first center c_1 uniformly at random in $\mathbf{X}$ and sets the arrival rate of each process $P_t(x)$ to be $\lambda_t(x) = \mathrm{cost}(x, \{c_1\})$. Then, it waits till one of the Poisson processes $P_t(x)$ jumps. When process $P_t(x)$ jumps, the algorithm adds the point $x \in \mathbf{X}$ (corresponding to that process) to the set of centers C_t and updates the arrival rates of all processes to be

$$\lambda_t(y) = \mathrm{cost}(y, C_t)$$

for all $y \in \mathbf{X}$. Note that if y is a center, then the arrival rate $\lambda_t(y)$ is 0.

The algorithm also maintains a round counter R. In the lazy version of this algorithm (which we describe in the next section), the algorithm makes a pass over the data set and samples a new batch of centers every time this counter is incremented. Additionally, at the end of each round, the algorithm checks if it chose at least one center in that round, and in the unlikely event that it did not, it selects one center with probability proportional to the costs of the points.

Initially, the algorithm sets $R = 0$, $t_0 = 0$, and $t_1 = \ell/\mathrm{cost}(\mathbf{X}, \{c_1\})$. Then, at each time point t_i ($i \geq 1$), we increment R and compute

$$t_{i+1} = t_i + \ell/\mathrm{cost}(\mathbf{X}, C_{t_i}),$$

where C_{t_i} is the set of all centers selected before time t_i. We refer to the time frame $[t_{i-1}, t_i]$ for $i \geq 1$ as the i-th round. The algorithm stops when one of the following conditions holds true (1) the

number of sampled centers is k; or (2) the round counter R equals the prespecified threshold R^*, which may be finite or infinite.

Before analyzing this algorithm, we mention that every Poisson process P_t with a variable arrival rate λ_t can be coupled with a Poisson process Q_s with rate 1. To this end, we substitute the variable

$$s(t) = \int_0^t \lambda_\tau d\tau,$$

and let

$$P_t \equiv Q_{s(t)}.$$

Observe that the expected number of arrivals for process Q_s in the infinitesimal interval $[s, s+ds]$ is $ds = \lambda_t dt$ which is exactly the same as for process P_t.

It is convenient to think about the variables s as "current position", t as "current time", and λ_t as "current speed" of s. To generate process $P_t(x)$, we can first generate Poisson process $Q_s(x)$ with arrival rate 1 and then move the position $s_t(x)$ with speed $\lambda_t(x)$. The process $P_t(x) = Q_{s_t(x)}(x)$ is a Poisson process with variable arrival rate $\lambda_t(x)$.

Theorem 7.1. *I. If the number of rounds is not bounded (i.e., $R^* = \infty$), then the distribution of centers returned by k-means++$_{ER}$ is identical to the distribution of centers returned by k-means++.*

II. Moreover, the expected number of rounds R is upper bounded by

$$(1 + o_k(1)) \cdot \left(\lceil \frac{k}{\ell} \rceil + \log \frac{2\,\mathrm{OPT}_1(\mathbf{X})}{\mathrm{OPT}_k(\mathbf{X})} \right),$$

and never exceeds k.

III. If the threshold R^* is given ($R^* < \infty$), then the cost of the solution after R^* rounds is upper bounded by

$$5(\ln k + 2)\,\mathrm{OPT}_k(\mathbf{X}) + 2R^* \left(\frac{4k}{e\ell R^*} \right)^{R^*} \cdot \mathrm{OPT}_1(\mathbf{X}).$$

Proof of Part I. For the sake of analysis, we assume that after the algorithm outputs solution C, it does not terminate, but instead continues to simulate Poisson processes $P_t(x)$. It also continues to update the set C_t (but, of course, not the solution) and the arrival rates $\lambda_t(x)$ till the set C_t contains k centers. Once $|C_t| = k$, the algorithm stops updating the set of centers C_t and arrival rates but still simulates continuous-time processes $P_t(x)$. Clearly, this additional phase of the algorithm does not affect the solution since it starts after the solution is already returned to the user.

We prove by induction on i that the first i centers $c_1, \ldots, c_i$ have exactly the same joint distribution as in k-means++. Indeed, the first center c_1 is drawn uniformly at random from the data set $\mathbf{X}$ as in k-means++. Suppose centers $c_1, \ldots, c_i$ are already selected. Then, we choose the next center c_{i+1} at the time of the next jump of one of the Poisson processes $P_t(x)$. Observe that the conditional probability that a particular process $P_t(x)$ jumps given that one of the processes $P_t(y)$ ($y \in \mathbf{X}$) jumps is proportional to $\lambda_t(x)$, which in turn equals the current $\mathrm{cost}(x, C_t)$ of point x. Hence, the distribution of center c_{i+1} is the same as in k-means++. This completes the proof of item I. □

Proof of Part II. We now show items II and III. Define process

$$P_t(\mathbf{X}) = \sum_{x \in \mathbf{X}} P_t(x).$$

Its rate $\lambda_t(\mathbf{X})$ equals $\sum_{x\in\mathbf{X}} \lambda_t(x)$. We couple this process with a Poisson $Q_s(\mathbf{X})$ with arrival rate 1 as discussed above. We want to estimate the number of centers chosen by the algorithm in the first R' rounds. To this end, we count the number of jumps of the Poisson process $P_t(\mathbf{X})$ (recall that we add a new center to C_t whenever $P_t(\mathbf{X})$ jumps unless $|C_t|$ already contains k centers). The number of jumps equals $P_{t_{R'}}$ which, in turn, equals $Q_{s_{R'}}$ where $s_{R'}(\mathbf{X})$ is the position of $s(\mathbf{X})$ at time $t_{R'}$:

$$s_{R'}(\mathbf{X}) = \int_0^{t_{R'}} \lambda_\tau(\mathbf{X})\, d\tau = \sum_{i=0}^{R'-1} \int_{t_i}^{t_{i+1}} \lambda_\tau(\mathbf{X})\, d\tau \geq \sum_{i=0}^{R'-1} (t_{i+1}-t_i)\cdot \lambda_{t_{i+1}}(\mathbf{X}).$$

Here, we used that $\lambda_t(\mathbf{X})$ is non-increasing, and thus, $\lambda_{t_{i+1}}(\mathbf{X}) \leq \lambda_\tau(\mathbf{X})$ for all $\tau \in [t_i, t_{i+1}]$. We now recall that $(t_{i+1}-t_i) = \ell/\mathrm{cost}(\mathbf{X}, C_{t_i})$ and $\lambda_{t_{i+1}}(\mathbf{X}) = \mathrm{cost}(\mathbf{X}, C_{t_{i+1}})$. Hence,

$$s_{R'}(\mathbf{X}) \geq \ell \sum_{i=0}^{R'-1} \frac{\mathrm{cost}(\mathbf{X}, C_{t_{i+1}})}{\mathrm{cost}(\mathbf{X}, C_{t_i})}.$$

By the inequality of arithmetic and geometric means, we have

$$\begin{aligned} s_{R'}(\mathbf{X}) &\geq \ell \cdot R' \left(\prod_{i=0}^{R'-1} \frac{\mathrm{cost}(\mathbf{X}, C_{t_{i+1}})}{\mathrm{cost}(\mathbf{X}, C_{t_i})} \right)^{1/R'} = \ell \cdot R' \left(\frac{\mathrm{cost}(\mathbf{X}, C_{t_{R'}})}{\mathrm{cost}(\mathbf{X}, C_{t_0})} \right)^{1/R'} \\ &= \ell \cdot R' \left(\frac{\mathrm{cost}(\mathbf{X}, C_{t_{R'}})}{\mathrm{cost}(\mathbf{X}, \{c_1\})} \right)^{1/R'}. \end{aligned} \tag{5}$$

We now use this equation to prove items II and III. For item II, we let random variable R' to be

$$R' = 2e\lceil k/\ell \rceil + \log \frac{\mathrm{cost}(\mathbf{X}, \{c_1\})}{\mathrm{OPT}_k(\mathbf{X})}.$$

Note that R' depends on the first center c_1 (which is chosen in the very beginning of the algorithm) but not on the Poisson processes $P_t(x)$. Since, C_t always contains at most k centers, we have $\mathrm{cost}(x, C_{t_{R'}}) \geq \mathrm{OPT}_k(\mathbf{X})$, and consequently

$$s_{R'}(\mathbf{X}) \geq \ell \cdot R' \left(\frac{\mathrm{OPT}_k(\mathbf{X})}{\mathrm{cost}(\mathbf{X}, \{c_1\})} \right)^{1/R'} > \ell \cdot 2e\lceil k/\ell \rceil \cdot 1/e \geq 2k.$$

The expected number of jumps of the Poisson process $Q_s(\mathbf{X})$ in the interval $[0, s_{R'}(\mathbf{X})]$ equals $Q_{s_R(\mathbf{X})}(\mathbf{X})$. Observe that

$$Q_{s_R(\mathbf{X})}(\mathbf{X}) \geq Q_{2k}(\mathbf{X})$$

and $Q_{2k}(\mathbf{X})$ is a Poisson random variable with parameter $2k$. By the Chernoff bound[2], it makes fewer than k jumps with exponentially small probability in k; namely, with probability at most $(e/2)^{-k}$. Thus, with probability at least $1-(e/2)^{-k}$, the algorithm selects k centers in the first R' rounds. Moreover, if it does not happen in the first R^* rounds, then it selects k centers by the end of the second R' rounds again with probability at least $1-(e/2)^{-k}$ and so on. Hence, the expected

[2]We use the bound $\Pr\{P \leq k\} \leq e^{-\lambda}\big(e\lambda/k\big)^k$, where P is a Poisson random variable with parameter λ and $k < \lambda$. See e.g., Theorem 5.4.2 in Mitzenmacher and Upfal (2017).

number of rounds till it selects k centers is $(1+o_k(1))R'$. Finally, observe that the expectation of $\mathrm{cost}(\mathbf{X},\{c_1\})$ over the choice of the first center equals $2\,\mathrm{OPT}_k(\mathbf{X})$. Since $\log(\cdot)$ is a convex function, we have

$$\mathbb{E}[R'] \le 2e\lceil k/\ell\rceil + \log\frac{2\,\mathrm{OPT}_1(\mathbf{X})}{\mathrm{OPT}_k(\mathbf{X})}.$$

Therefore, we showed that the expected number of rounds is upper bounded by the right hand side of the expression above times a multiplicative factor of $(1+o_k(1))$. A slightly more careful analysis gives a bound of

$$(1+o_k(1))\left(e\lceil k/\ell\rceil + \log\frac{2\,\mathrm{OPT}_1(\mathbf{X})}{\mathrm{OPT}_k(\mathbf{X})}\right).$$

This concludes the proof of item II. □

Proof of Part III. We now prove item III. Denote $T = t_{R^*}$. Consider the event

$$\mathcal{E} = \{\text{algorithm samples } k \text{ centers in the first } R^* \text{ rounds}\}.$$

Let $\bar{\mathcal{E}}$ be the complimentary events to $\mathcal{E}$. Then,

$$\mathbb{E}\big[\mathrm{cost}(\mathbf{X},C_T)\big] = \mathbb{E}\big[\mathrm{cost}(\mathbf{X},C_T)\cdot\mathbb{1}(\mathcal{E})\big] + \mathbb{E}\big[\mathrm{cost}(\mathbf{X},C_T)\cdot\mathbb{1}(\bar{\mathcal{E}})\big].$$

We now separately upper bound each of the terms on the right hand side. It is easy to upper bound the first term:

$$\mathbb{E}[\mathrm{cost}(\mathbf{X},C_T)\cdot\mathbb{1}(\mathcal{E})] \le 5(\ln k+2)\cdot\mathrm{OPT}_k(\mathbf{X}),$$

because the distribution of centers returned by k-means++$_{\mathrm{ER}}$ is identical to the distribution of centers returned by k-means++. We now bound the second term. Denote by $\mathcal{D}_\rho$ the event

$$\mathcal{D}_\rho = \left\{\mathrm{cost}(\mathbf{X},C_T) \ge \left(\frac{\rho k}{\ell R^*}\right)^{R^*}\mathrm{cost}(\mathbf{X},\{c_1\})\right\}.$$

We prove the following claim.

Claim 7.2. *The following inequality holds for every real number $\rho \in [1, \ell R^*/k]$ and any choice of the first center c_1:*

$$\Pr\left(\bar{\mathcal{E}} \text{ and } \mathcal{D}_\rho \mid c_1\right) \le e^{-(\rho-1)k}\rho^{k-1}.$$

Proof. We use inequality (5) with $R' = R^*$:

$$s_{R^*}(\mathbf{X}) \ge \ell\cdot R^*\left(\frac{\mathrm{cost}(\mathbf{X},C_T)}{\mathrm{cost}(\mathbf{X},\{c_1\})}\right)^{1/R^*}.$$

It implies that $s_{R^*}(\mathbf{X}) \ge \rho k$ if event $\mathcal{D}_\rho$ occurs. On the other hand if $\bar{\mathcal{E}}$ occurs, then the number of centers chosen by the end of round R^* is less than k and, consequently, the number of jumps of $P_t(\mathbf{X})$ in the interval $[0,T]$ is less than k:

$$P_T(\mathbf{X}) \equiv Q_{s_{R^*}(\mathbf{X})}(\mathbf{X}) < k.$$

Hence, we can bound $\Pr(\bar{\mathcal{E}} \text{ and } \mathcal{D}_\rho \mid c_1)$ as follows:

$$\Pr(\bar{\mathcal{E}} \text{ and } \mathcal{D}_\rho) \le \Pr\left(\mathcal{D}_\rho \text{ and } Q_{s_{R^*}}(\mathbf{X}) < k \mid c_1\right) \le \\ \le \Pr\left(\mathcal{D}_\rho \text{ and } Q_{\rho k}(\mathbf{X}) < k \mid c_1\right) \le \Pr\left(Q_{\rho k}(\mathbf{X}) < k \mid c_1\right).$$

Random variable $Q_{\rho k}(\mathbf{X})$ has the Poisson distribution with parameter ρk and is independent of c_1. By the Chernoff bound, the probability that $Q_{\rho k}(\mathbf{X}) \le k-1$ is at most (as in Part II of the proof):

$$\Pr\left\{Q_{\rho k}(\mathbf{X}) \le k-1\right\} \le e^{-\rho k}\Big(\frac{e\rho k}{k-1}\Big)^{k-1} = e^{-(\rho-1)k-1}\rho^{k-1} \cdot \underbrace{\Big(\frac{k}{k-1}\Big)^{k-1}}_{\le e} \le e^{-(\rho-1)k}\rho^{k-1}.$$

This completes the proof of Claim 7.2. □

Let

$$Z = \left(\frac{\ell R^*}{k}\right)^{R^*} \cdot \frac{\text{cost}(\mathbf{X}, C_T)}{\text{cost}(\mathbf{X}, \{c_1\})}.$$

Then, by Claim 7.2,

$$\Pr\left(\bar{\mathcal{E}} \text{ and } Z \ge \rho^{R^*} \mid c_1\right) \le e^{-(\rho-1)k}\rho^{k-1}. \tag{6}$$

Write,

$$\mathbb{E}\left[\mathbb{1}(\bar{\mathcal{E}}) \cdot Z \mid c_1\right] = \int_0^\infty \Pr\left(\mathbb{1}(\bar{\mathcal{E}}) \text{ and } Z \ge r \mid c_1\right) dr \le 1 + \int_1^\infty \Pr\left(\mathbb{1}(\bar{\mathcal{E}}) \text{ and } Z \ge r \mid c_1\right) dr.$$

We now substitute $r = \rho^{R^*}$ and then use (6):

$$\mathbb{E}\left[Z \cdot \mathbb{1}(\bar{\mathcal{E}}) \mid c_1\right] \le 1 + R^* \int_1^\infty \Pr\left(\bar{\mathcal{E}} \text{ and } Z \ge \rho^{R^*} \mid c_1\right) \cdot \rho^{R^*-1} d\rho \\ \le 1 + R^* \int_1^\infty e^{-(\rho-1)k}\rho^{k+R^*-2} d\rho.$$

We note that $R^* < k$, since our algorithm chooses at least one center in each round. Thus, by Lemma 7.3 (which we prove below), the integral on the right hand side is upper bounded by ${}^{eR^*}\!/_2 \cdot ({}^4\!/_e)^{R^*}$. Hence,

$$\mathbb{E}\left[Z \cdot \mathbb{1}(\bar{\mathcal{E}}) \mid c_1\right] \le 1 + R^* \cdot \left(\frac{4}{e}\right)^{R^*-2}.$$

Multiplying both sides of the inequality by $({}^k\!/_{\ell R^*})^{R^*} \cdot \text{cost}(\mathbf{X}, \{c_1\})$ and taking the expectation over c_1, we get the desired inequality:

$$\mathbb{E}\left[\text{cost}(\mathbf{X}, C_T) \cdot \mathbb{1}(\bar{\mathcal{E}})\right] \le \left(1 + R^* \left(\frac{4}{e}\right)^{R^*}\right)\Big(\frac{k}{\ell R^*}\Big)^{R^*} \mathbb{E}_{c_1}\left[\text{cost}(\mathbf{X}, \{c_1\}\right] \\ = \left(1 + R^* \left(\frac{4}{e}\right)^{R^*-2}\right)\Big(\frac{k}{\ell R^*}\Big)^{R^*} \cdot 2\,\text{OPT}_1(\mathbf{X}) \\ < 2R^* \left(\frac{4k}{e\ell R^*}\right)^{R^*} \text{OPT}_1(\mathbf{X}).$$

This finishes the proof of Theorem 7.1. □

Lemma 7.3. *For $R^* < k$, we have*

$$\int_1^\infty e^{-(\rho-1)k}\rho^{k+R^*-2}d\rho \le \frac{e}{2}\left(\frac{4}{e}\right)^{R^*}.$$

Proof. Since $e^{-(\rho-1)}\rho \le 1$ for all $\rho \ge 1$, we have $e^{-(\rho-1)k}\rho^k \le e^{-(\rho-1)R^*}\rho^{R^*}$ for any $R^* < k$. Thus, we have

$$\begin{aligned}\int_1^\infty e^{-(\rho-1)k}\rho^{k+R^*-2}d\rho &\le \int_1^\infty e^{-(\rho-1)R^*}\rho^{2R^*-3}d\rho = e^{R^*}\int_1^\infty e^{-\rho R^*}\rho^{2R^*-3}d\rho \\ &= e^{R^*}\int_1^\infty (e^{-\rho}\rho^2)^{R^*}\rho^{-3}d\rho.\end{aligned}$$

Observe that $e^{-\rho}\rho^2 \le 4/e^2$ for any $\rho \ge 1$. Hence,

$$(e^{-\rho}\rho^2)^{R^*} = (e^{-\rho}\rho^2)^{R^*-1}\cdot e^{-\rho}\rho^2 \le (4/e^2)^{R^*-1}e^{-\rho}\rho^2.$$

Thus,

$$\int_1^\infty e^{-(\rho-1)k}\rho^{k+R^*-2}d\rho \le \frac{4^{R^*-1}\cdot e^{R^*}}{e^{2(R^*-1)}}\cdot\int_1^\infty \frac{e^{-\rho}}{\rho}\,d\rho = \frac{4^{R^*-1}}{e^{R^*-2}}\cdot\frac{1}{4} = \left(\frac{4}{e}\right)^{R^*-2}.$$

□

7.2 Lazy implementation of k-means++$_{\mathbf{ER}}$

We now describe how we can efficiently implement the k-means++$_{\mathrm{ER}}$ algorithm using a lazy reservoir sampling. We remind the reader that the time of the first jump of a Poisson process with parameter λ is distributed as the exponential distribution with parameter λ. Imagine for a moment, that the arrival rates of our Poisson processes were constant. Then, in order to select the first k jumps, we would generate independent exponential random variables with parameters $\lambda(x)$ for all x and choose k smallest values among them. This algorithm is known as the reservoir sampling(see Efraimidis and Spirakis (2006)). To adapt this algorithm to our needs, we need to update the arrival rates of the exponential random variables. Loosely speaking, we do so by generating exponential random variables with rate 1 for Poisson processes $Q_s(x)$ which are described above and then updating the speeds $\lambda_t(x)$ of variables $s_t(x)$. We now formally describe the algorithm.

In the beginning of every round i, we recompute costs of all points in the data set. Then, we draw an independent exponential random variable $\mathcal{S}_x$ with rate 1 for every point x, and let $S_t(x) = \mathcal{S}_x$. We set

$$\tau_t(x) = \frac{S_t(x)}{\lambda_t(x)}.$$

Think of $S_t(x)$ as the distance $s_t(x)$ needs to travel till process $Q_s(x)$ jumps; $\lambda_t(x)$ is the speed of point $s_t(x)$; and $\tau_t(x)$ is the time left till $Q_s(x) = P_t(x)$ jumps if the speed λ_t does not change. Among all points $x \in X$, we select a tentative set of centers Z for this round. The set Z contains all points x with $t_{i-1} + \tau_t(x) \le t_i$. This is the set of all points for which their Poisson processes would jump in the current round if their arrival rates remained the same till the end of the round. Since the arrival rates can only decrease in our algorithm, we know for sure that for points x outside of Z, the corresponding processes $P_t(x)$ will not jump in this round. Thus, we can safely ignore those points during the current round.

We also note that in the unlikely event that the initial set Z is empty, we choose x with the smallest time $\tau_t(x)$ and add it to the set of centers C_t. (This is equivalent to choosing a point with probability proportional to $\text{cost}(x, C_t)$ by the memorylessness property of the exponential distribution).

The steps we described above – updating costs $\text{cost}(x, C_t)$, drawing exponential random variables $\mathcal{S}_x$, and selecting points in the set Z – can be performed in parallel using one pass over the data set. In the rest of the current round, our algorithm deals only with the set Z whose size in expectation is at most ℓ (see below).

While the set Z is not empty we do the following. We choose $x \in Z$ with the smallest value of $\tau_t(x)$. This x corresponds to the process that jumps first. Then, we perform the following updates: We add x to the set of centers C_t. We set the "current time" t to $t = t' + \tau_{t'}(x)$, where t' is the time of the previous update. If x is the first center selected in the current round, then we let t' to be the time when the round started (i.e., t_{i-1}). We recompute the arrival rates (speeds) $\lambda_t(x)$ for each x in Z. Finally, we update the values of all $\tau_t(x)$ for $x \in Z$ using the formula

$$\tau_t(x) = \frac{S_t(x) - \lambda_{t'}(x) \cdot (t - t')}{\lambda_t(x)},$$

here $\lambda_{t'}(x) \cdot (t - t')$ is the distance variable $s_t(x)$ moved from the position where it was at time t'; $S_t(x) - \lambda_{t'}(x) \cdot (t - t')$ is the remaining distance $s_t(x)$ needs to travel till the process $Q_t(x)$ jumps; and $\tau_t(x)$ is the remaining time till $P_t(x)$ jumps if we do not update its arrival rate. After we update $\tau_t(x)$, we prune the set Z. Specifically, we remove from set Z all points x with $t + \tau_t(x) > t_i$. As before, we know for sure that if x is removed from Z, then the corresponding processes $P_t(x)$ will not jump in the current round.

This algorithm simulates the process we described in the previous section. The key observation is that Poisson processes $P_t(x)$ we associate with points x removed from Z cannot jump in this round and thus can be safely removed from our consideration. We now show that the expected size of the set Z is at most ℓ. In the next section, we analyze the running time of this algorithm.

Then we show that the expected size of the set Z in the beginning of each round $i + 1$ is at most ℓ. Since every point x belongs to Z with probability

$$\Pr\{x \in Z\} = \Pr\left\{ \frac{\mathcal{S}_x}{\text{cost}(x, C_{t_i})} \le \frac{\ell}{\text{cost}(\mathbf{X}, C_{t_i})} \right\} = \Pr\left\{ \mathcal{S}_x \le \ell \cdot \frac{\text{cost}(x, C_{t_i})}{\text{cost}(\mathbf{X}, C_{t_i})} \right\}.$$

The right hand side is the probability that the Poisson process $Q_s(x)$ with rate 1 jumps in the interval of length $\ell \cdot \text{cost}(x, C_{t_i})/\text{cost}(\mathbf{X}, C_{t_i})$ which is upper bounded by the expected number of jumps of $Q_s(x)$ in this interval. The expected number of jumps exactly equals $\ell \cdot \text{cost}(x, C_{t_i})/\text{cost}(\mathbf{X}, C_{t_i})$. Thus, the expected size of Z is upper bounded as

$$\mathbb{E}|Z| = \sum_{z \in \mathbf{X}} \Pr\{z \in Z\} \le \sum_{z \in \mathbf{X}} \ell \cdot \frac{\text{cost}(z, C_{t_i})}{\text{cost}(\mathbf{X}, C_{t_i})} = \ell.$$

7.3 Run time analysis

According to our analysis above, the number of new centers chosen at each round of k-means++$_{\text{ER}}$ is at most the size of set Z, which is $O(\ell)$ with high probability. In the beginning of every round, we need to update costs of all data points, which requires $O(n\ell d)$ time. In each round, we also

need to maintain the rates of all points in set Z, which needs $O(\ell^2 d)$ time. Thus, the total running time for k-means++$_{\mathrm{ER}}$ with R rounds is $O(Rn\ell d)$. We note that before running our algorithm, we can reduce the dimension d of the space to $O(\log k)$ using the Johnson–Lindenstrauss transform (see Johnson and Lindenstrauss (1984)). This will increase the approximation factor by a factor of $(1+\varepsilon)$ but make the algorithm considerably faster (see Makarychev et al. (2019), Becchetti et al. (2019), and Boutsidis et al. (2010)).

References

A. Aggarwal, A. Deshpande, and R. Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 15–28. Springer, 2009.

S. Ahmadian, A. Norouzi-Fard, O. Svensson, and J. Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *SIAM Journal on Computing*, pages FOCS17–97, 2019.

D. Aloise, A. Deshpande, P. Hansen, and P. Popat. Np-hardness of euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.

D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

P. Awasthi, M. Charikar, R. Krishnaswamy, and A. K. Sinop. The hardness of approximation of euclidean k-means. In *31st International Symposium on Computational Geometry (SoCG 2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

O. Bachem, M. Lucic, and A. Krause. Distributed and provably good seedings for k-means in constant rounds. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 292–300. JMLR. org, 2017.

B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7):622–633, 2012.

L. Becchetti, M. Bury, V. Cohen-Addad, F. Grandoni, and C. Schwiegelshohn. Oblivious dimension reduction for k-means: beyond subspaces and the johnson–lindenstrauss lemma. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1039–1050, 2019.

B. Boehmke and B. M. Greenwell. *Hands-on machine learning with R*. CRC Press, 2019.

C. Boutsidis, A. Zouzias, and P. Drineas. Random projections for k-means clustering. In *Advances in Neural Information Processing Systems*, pages 298–306, 2010.

T. Brunsch and H. Röglin. A bad instance for k-means++. *Theoretical Computer Science*, 505: 19–26, 2013.

D. Choo, C. Grunau, J. Portmann, and V. Rozhoň. k-means++: few more steps yield constant approximation. In *Proceedings of the 37th International Conference on Machine Learning*, pages 7849–7057. JMLR. org, 2020.

S. Dasgupta. *The hardness of k-means clustering*. Department of Computer Science and Engineering, University of California, San Diego, 2008.

S. Dasgupta. UCSD CSE 291, Lecture Notes: Geometric Algorithms, 2013. URL: https://cseweb.ucsd.edu/~dasgupta/291-geom/kmeans.pdf. Last visited on 2020/06/01.

D. Dua and C. Graff. UCI ML repository, 2017. URL http://archive.ics.uci.edu/ml.

P. S. Efraimidis and P. G. Spirakis. Weighted random sampling with a reservoir. *Information Processing Letters*, 97(5):181–185, 2006.

R. Elber. Kdd-Cup, 2004. URL http://osmot.cs.cornell.edu/kddcup/.

W. B. Johnson and J. Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. A local search approximation algorithm for k-means clustering. *Computational Geometry*, 28(2): 89 – 112, 2004. ISSN 0925-7721. doi: https://doi.org/10.1016/j.comgeo.2004.03.003. URL http://www.sciencedirect.com/science/article/pii/S0925772104000215.

S. Lattanzi and C. Sohler. A better k-means++ algorithm via local search. In *International Conference on Machine Learning*, pages 3662–3671, 2019.

E. Lee, M. Schmidt, and J. Wright. Improved and simplified inapproximability for k-means. *Information Processing Letters*, 120:40–43, 2017.

S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

K. Makarychev, Y. Makarychev, M. Sviridenko, and J. Ward. A bi-criteria approximation algorithm for k-means. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2016.

K. Makarychev, Y. Makarychev, and I. Razenshteyn. Performance of johnson–lindenstrauss transform for k-means and k-medians clustering. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1027–1038, 2019.

M. Mitzenmacher and E. Upfal. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

R. Ostrovsky, Y. Rabani, L. Schulman, and C. Swamy. The effectiveness of lloyd-type methods for the k-means problem. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, 2006.

V. Rozhoň. Simple and sharp analysis of k-means||. In *Proceedings of the 37th International Conference on Machine Learning*, pages 7828–7837. JMLR. org, 2020.

D. Wei. A constant-factor bi-criteria approximation guarantee for k-means++. In *Advances in Neural Information Processing Systems*, pages 604–612, 2016.

Appendix

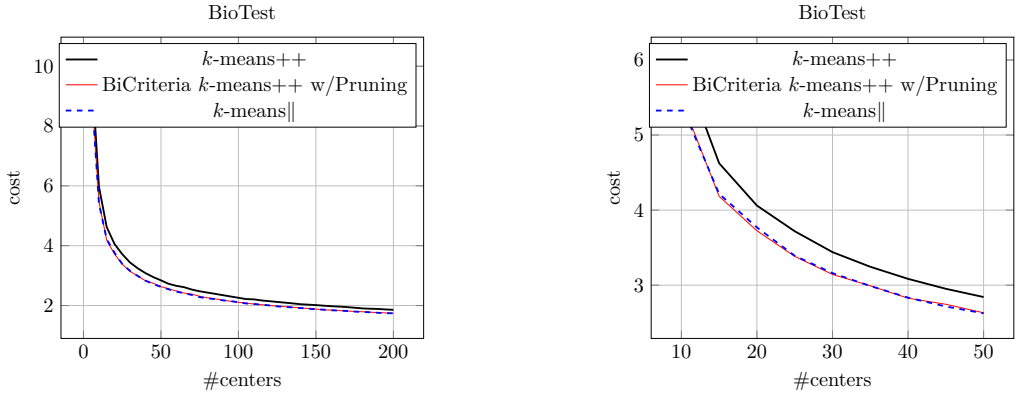
In this appendix, we present our experiments, give proofs omitted in the main part of the paper, and provide complimentary lower bounds.

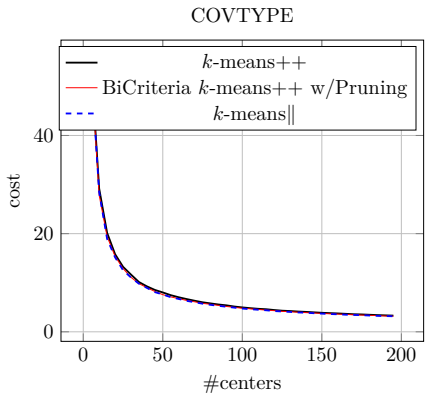
A Experiments

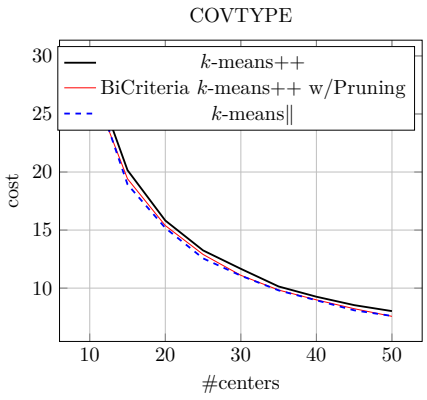
In this section, we present plots that show that the performance of k-means$\|$ and "k-means++ with oversampling and pruning" algorithms are very similar in practice. Below, we compare the following algorithms on the datasets BioTest from KDD Cup 2004 (Elber, 2004) and COVTYPE from the UCI ML repository (Dua and Graff, 2017):

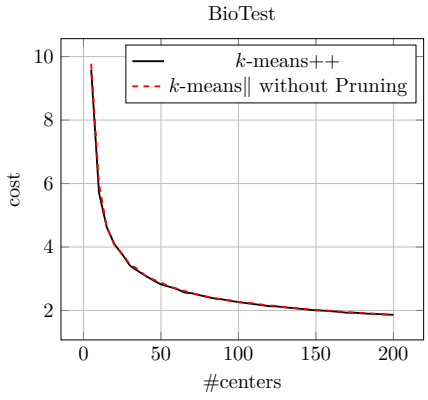
- Regular k-means++. The performance of this algorithm is shown with a solid black line on the plots below.
- k-means$\|$ without pruning. This algorithm samples k centers using k-means$\|$ with $T = 5$ rounds and $\ell = k/T$.
- k-means$\|$. This algorithm first samples $5k$ centers using k-means$\|$ and then subsamples k centers using k-means++. The performance of this algorithm is shown with a dashed blue line on the plots below.
- k-means++ with oversampling and pruning. This algorithm first samples $5k$ centers using k-means++ and then subsamples k centers using k-means++. The performance of this algorithm is shown with a thin red line on the plots below.

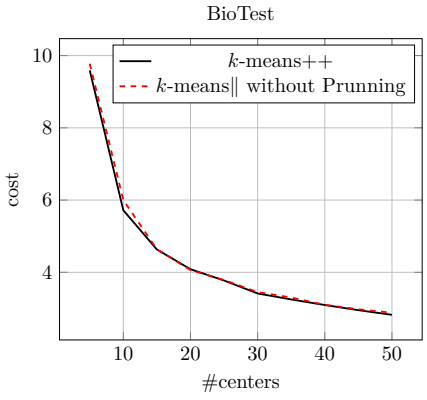
For each $k = 5, 10, \cdots, 200$, we ran these algorithms for 50 iterations and took their average. We normalized all costs by dividing them by the cost of k-means++ with $k = 1000$ centers.

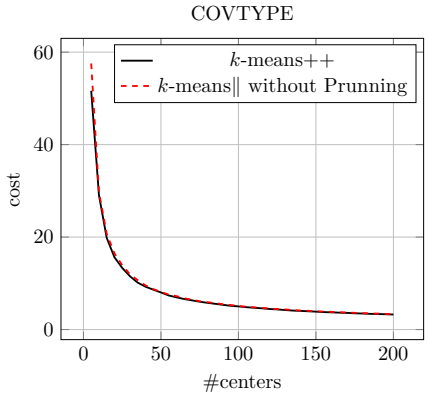


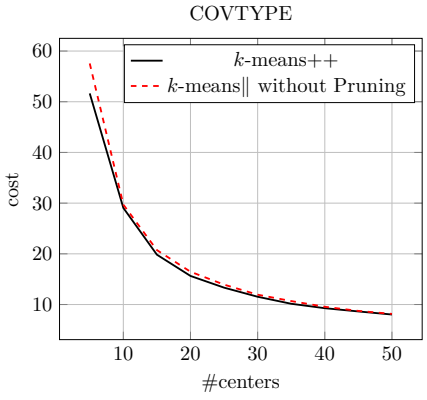

COVTYPE
k-means++
BiCriteria k-means++ w/Pruning
k-means‖
cost
#centers
0
20
40
0
50
100
150
200


COVTYPE
k-means++
BiCriteria k-means++ w/Pruning
k-means‖
cost
#centers
10
15
20
25
30
10
20
30
40
50


BioTest
k-means++
k-means‖ without Pruning
cost
#centers
2
4
6
8
10
0
50
100
150
200


BioTest
k-means++
k-means‖ without Prunning
cost
#centers
4
6
8
10
10
20
30
40
50


COVTYPE
k-means++
k-means‖ without Prunning
cost
#centers
0
20
40
60
0
50
100
150
200


COVTYPE
k-means++
k-means‖ without Pruning
cost
#centers
10
20
30
40
50
60
10
20
30
40
50

B Details for Preliminaries

For any set of points $\mathbf{Y} \subset \mathbb{R}^d$, let $\mu = \sum_{x \in \mathbf{Y}} x / |\mathbf{Y}|$ be the *centroid* of the cluster $\mathbf{Y}$. Then, the optimal cost of $\mathbf{Y}$ with one center,

$$\mathrm{OPT}_1(\mathbf{Y}) = \sum_{x \in \mathbf{Y}} \|x - \mu\|^2 = \frac{\sum_{(x,y) \in \mathbf{Y} \times \mathbf{Y}} \|x - y\|^2}{2\,|\mathbf{Y}|}.$$

This is a well known formula which is often used for analyzing of k-means algorithms. For completeness, we give a proof below.

Proof. Consider any point $z \in \mathbb{R}^d$, then we have:

$$\begin{aligned}
\mathrm{cost}(\mathbf{Y}, \{z\}) &= \sum_{x \in \mathbf{Y}} \|x - z\|^2 = \sum_{x \in \mathbf{Y}} \|(x - \mu) + (\mu - z)\|^2 \\
&= \sum_{x \in \mathbf{Y}} \left(\|x - \mu\|^2 + \|\mu - z\|^2 + 2\,\langle x - \mu, \mu - z \rangle \right) \\
&= \sum_{x \in \mathbf{Y}} \|x - \mu\|^2 + |\mathbf{Y}| \cdot \|\mu - z\|^2 + 2 \left\langle \sum_{x \in \mathbf{Y}} (x - \mu), \mu - z \right\rangle \\
&= \sum_{x \in \mathbf{Y}} \|x - \mu\|^2 + |\mathbf{Y}| \cdot \|\mu - z\|^2.
\end{aligned}$$

Thus, the optimal choice of z to minimize $\mathrm{cost}(\mathbf{Y}, \{z\})$ is μ and $\mathrm{OPT}_1(\mathbf{Y}) = \sum_{x \in \mathbf{Y}} \|x - \mu\|^2$.

$$\begin{aligned}
\sum_{x \in \mathbf{Y}} \|x - \mu\|^2 &= \sum_{x \in \mathbf{Y}} \langle x - \mu, x - \mu \rangle = \sum_{x \in \mathbf{Y}} \langle x, x - \mu \rangle \\
&= \sum_{x \in \mathbf{Y}} \left\langle x, x - \sum_{y \in \mathbf{Y}} \frac{y}{|\mathbf{Y}|} \right\rangle = \frac{1}{|\mathbf{Y}|} \sum_{(x,y) \in \mathbf{Y} \times \mathbf{Y}} \langle x, x - y \rangle \\
&= \frac{1}{2\,|\mathbf{Y}|} \left(\sum_{(x,y) \in \mathbf{Y} \times \mathbf{Y}} \langle x, x - y \rangle + \sum_{(x,y) \in \mathbf{Y} \times \mathbf{Y}} \langle y, y - x \rangle \right) \\
&= \frac{\sum_{(x,y) \in \mathbf{Y} \times \mathbf{Y}} \|x - y\|^2}{2\,|\mathbf{Y}|}.
\end{aligned}$$

□

C Lower bounds

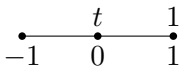
C.1 Lower bound on the cost of covered clusters

We show the following lower bound on the expected cost of a covered cluster in k-means++. Therefore, the 5-approximation in Lemma 4.1 is tight.

Theorem C.1. *For any $\varepsilon > 0$, there exists an instance of k-means such that for a set $P \in \mathbf{X}$ and a set of centers $C \in \mathbb{R}^d$, if a new center c is sampled from P with probability $\Pr(c = x) = \mathrm{cost}(x, C)/\mathrm{cost}(P, C)$, then*

$$\mathbb{E}_c \left[\mathrm{cost}(P, C \cup \{c\}) \right] \geq (5 - \varepsilon) \mathrm{OPT}_1(P).$$

Proof. Consider the following one dimensional example, where P contains t points at 0 and one point at 1, and the closest center already chosen in C to P is at -1.



The new center c will be chosen at 0 with probability $\frac{t}{t+4}$, and at 1 with probability $\frac{4}{t+4}$. Then, the expected cost of P is

$$\mathbb{E}_c\left[\text{cost}(P, C \cup \{c\})\right] = 1 \cdot \frac{t}{t+4} + t \cdot \frac{4}{t+4} = \frac{5t}{t+4};$$

and the optimal cost of P is $\text{OPT}_1(P) \leq 1$. Thus, by choosing $t \geq 4(5-\varepsilon)/\varepsilon$, we have

$$\mathbb{E}_c\left[\text{cost}(P, C \cup \{c\})\right] \geq (5-\varepsilon)\text{OPT}_1(P).$$

□

C.2 Lower bound on the bi-criteria approximation

In this section, we show that the bi-criteria approximation bound of $O(\ln \frac{k}{\Delta})$ is tight up to constant factor. Our proof follows the approach by Brunsch and Röglin (2013). We show the following theorem.

Theorem C.2. *For every $k > 1$ and $\Delta \leq k$, there exists an instance $\mathbf{X}$ of k-means such that the bi-criteria k-means++ algorithm with $k + \Delta$ centers returns a solution of cost greater than*

$$\frac{1}{8} \log \frac{k}{\Delta} \cdot \text{OPT}_k(\mathbf{X})$$

with probability at least $1 - e^{-\sqrt{k}/2}$.

Remark: This implies that the expected cost of bi-criteria k-means with $k + \Delta$ centers is at least

$$\frac{1 - e^{-\sqrt{k}/2}}{8} \cdot \log \frac{k}{\Delta} \cdot \text{OPT}_k(\mathbf{X}).$$

Proof. For every k and $\Delta \geq \sqrt{k}$, we consider the following instance. The first cluster is a scaled version of the standard simplex with $N \gg k$ vertices centered at the origin, which is called the heavy cluster. The length of the edges in this simplex is $1/\sqrt{N-1}$. Each of the remaining $k-1$ clusters contains a single point on $k-1$ axes, which are called light clusters. These clusters are located at distance $\sqrt{\alpha}$ from the center of the heavy cluster and $\sqrt{2\alpha}$ from each other, where $\alpha = \frac{\ln(k/\Delta)}{4\Delta}$.

For the sake of analysis, let us run k-means++ till we cover all clusters. At the first step, the k-means++ algorithm almost certainly selects a center from the heavy cluster since $N \gg k$. Then, at each step, the algorithm can select a center either from one of uncovered light clusters or from the heavy cluster. In the former case, we say that the algorithm hits a light cluster, and in the latter case we say that the algorithm misses a light cluster. Below, we show that with high probability the algorithm makes at least 2Δ misses before it covers all but Δ light clusters.

Lemma C.3. *Let $\Delta \geq \sqrt{k}$. By the time the k-means++ algorithm covers all but Δ light clusters, it makes greater than 2Δ misses with probability at least $1 - e^{-\sqrt{k}/2}$.*

Proof sketch. Let $\varepsilon = 1/\sqrt{N}$. Observe that k-means++ almost certainly covers all clusters in εN steps (since $N \gg k$). So in the rest of this proof sketch, we assume that the number chosen centers is at most εN and, consequently, at least $(1-\varepsilon)N$ points in the heavy cluster are not selected as centers. Hence, the cost of the heavy cluster is at least $1 - \varepsilon$.

Consider a step of the algorithm when exactly u light clusters remain uncovered. At this step, the total cost of all light clusters is αu (we assume for simplicity that distance between the light clusters and the closest chosen center in the heavy cluster is the same as the distance to the origin). The cost of the heavy cluster is at least $1 - \varepsilon$. The probability that the algorithm chooses a center from the heavy cluster and thus misses a light cluster is at least $(1-\varepsilon)/(1+\alpha u)$.

Define random variables $\{X_u\}$ as follows. Let $X_u = 1$ if the algorithm misses a cluster at least once when the number of uncovered light clusters is u; and let $X_u = 0$, otherwise. Then, $\{X_u\}$ are independent Bernoulli random variables. For each u, we have $\mathbb{P}[X_u = 1] \geq (1-\varepsilon)/(1+\alpha u)$.

Observe that the total number of misses is lower bounded by $\sum_{u=\Delta}^{k-1} X_u$. Then, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{u=\Delta}^{k-1} X_u\right] &\geq (1-\varepsilon)\sum_{u=\Delta}^{k-1} \frac{1}{1+\alpha u} \geq (1-\varepsilon)\int_{\Delta}^{k} \frac{\mathrm{d}u}{1+\alpha u} \\
&= (1-\varepsilon)\alpha^{-1}\ln\frac{1+\alpha k}{1+\alpha\Delta} \\
&\geq (1-\varepsilon)\alpha^{-1}\ln\frac{k}{\Delta} = 4(1-\varepsilon)\Delta.
\end{aligned}$$

Let $\mu = \mathbb{E}\left[\sum_{u=\Delta}^{k-1} X_u\right] \geq 4(1-\varepsilon)\Delta$. By the Chernoff bound for Bernoulli random variables, we have

$$\mathbb{P}\left[\sum_{u=\Delta}^{k} X_u \leq 2\Delta\right] \leq e^{-\mu}\left(\frac{e\mu}{2\Delta}\right)^{2\Delta}.$$

Since $f(x) = e^{-x}(\frac{ex}{2\Delta})^{2\Delta}$ is a monotone decreasing function for $x \geq 2\Delta$, we have

$$\mathbb{P}\left[\sum_{u=\Delta}^{k} X_u \leq 2\Delta\right] \leq e^{-(2-4\varepsilon)\Delta}\cdot 2^{2\Delta} \leq e^{-\Delta/2}.$$

Hence, with probability as least $1 - e^{-\sqrt{k}/2}$, the number of misses is greater than 2Δ. □

For every k and $\Delta \geq \sqrt{k}$, consider the instance we constructed. By Lemma C.3, the algorithm chooses more than $k+\Delta$ centers to cover all but Δ light clusters with probability at least $1-e^{-\sqrt{k}/2}$. Thus, at the time when the algorithm chose $k+\Delta$ centers, the number of uncovered light clusters was greater than Δ. Hence, in the clustering with $k+\Delta$ centers sampled by k-means++, the total cost is at least $\frac{1}{4}\ln(k/\Delta)$, while the cost of the optimal solution with k clusters is 1. For every k and $\Delta < \sqrt{k}$, the total cost is at least $\frac{1}{4}\ln(k/\Delta')$ with $\Delta' = \sqrt{k}$ extra centers, which concludes the proof. □