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

Nithin Balaji S P, Pulkit R. Khandelwal
Industrial & Systems Engineering, University of Illinois, Urbana-Champaign
IE 529: Stats of Big Data and Clustering
Professor Carolyn Beck
May 8, 2024

Introduction:

Clustering algorithms play a critical role in data analysis, enabling the extraction of meaningful patterns from complex datasets across various industries. The problem centers around unsupervised learning, specifically clustering data into groups with shared characteristics by assessing the similarity between points in the cluster accuracy and algorithms empirical performance. The k-means algorithm (Lloyd's algorithm) is commonly employed, but its clustering accuracy is heavily dependent on the initial random selection of cluster centers. The primary objective of clustering algorithms is to minimize the sum of squared distances between points and their nearest cluster center. Consequently, enhancing clustering quality and improving approximation guarantees are key challenges, with new methods like k-means++ and k-means++ parallel addressing the limitations of traditional k-means.

This problem is significant because k-means clustering is a widely used unsupervised learning technique in data science and machine learning. As data generation grows exponentially, conventional clustering methods often lack efficiency and scalability. Improved initial cluster selection through enhanced algorithms like k-means++ and its parallel variant leads to more accurate clustering and optimizations. This enhancement is crucial, given that clustering underpins many applications in computer vision and data mining, where quality improvements yield better results.

To solve this problem, several assumptions are made. First, initial cluster centers are randomly chosen from the data points, which is crucial for theoretical guarantees. Similarity between data points is often determined using a metric space, with assumptions made on the choice of similarity function (e.g., Gaussian function or Euclidean distance). Furthermore, data used to validate these theoretical assumptions must be uniformly distributed, as biased data would undermine practical application.

Technical Background:

The assumptions made and the key concepts establish the groundwork to analyze the different clustering algorithms. For a given set of points $X = \{x_1, x_2, ...x_n\} \subseteq R^d$ and for k greater than equal to 1,

Objective function:

The k-means clustering problem is to find a set C of k centers in R^d to minimize

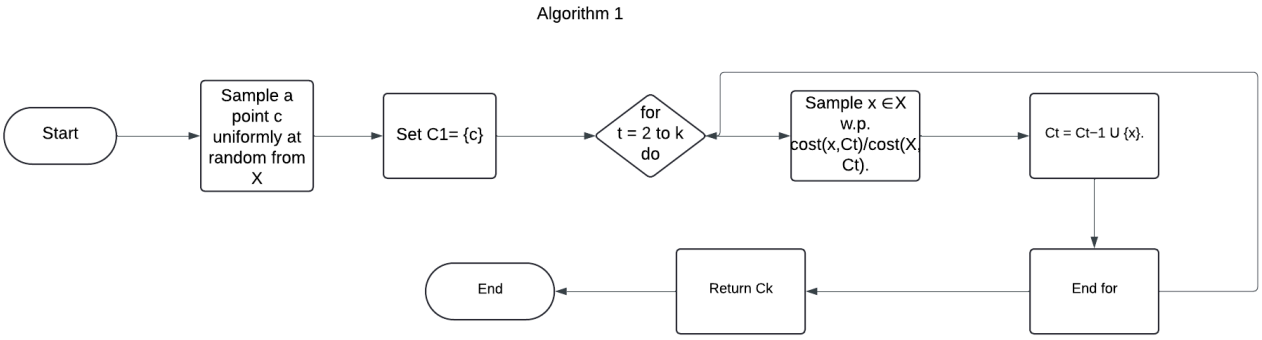
$$\text{cost}(\mathbf{X}, C) := \sum_{x \in \mathbf{X}} \min_{c \in C} \|x - c\|^2.$$

Optimal Cost for any X:

$$\text{OPT}_i(\mathbf{X}) := \min_{|C|=i} \text{cost}\,(\mathbf{X}, C)$$

K-means++ Algorithm:

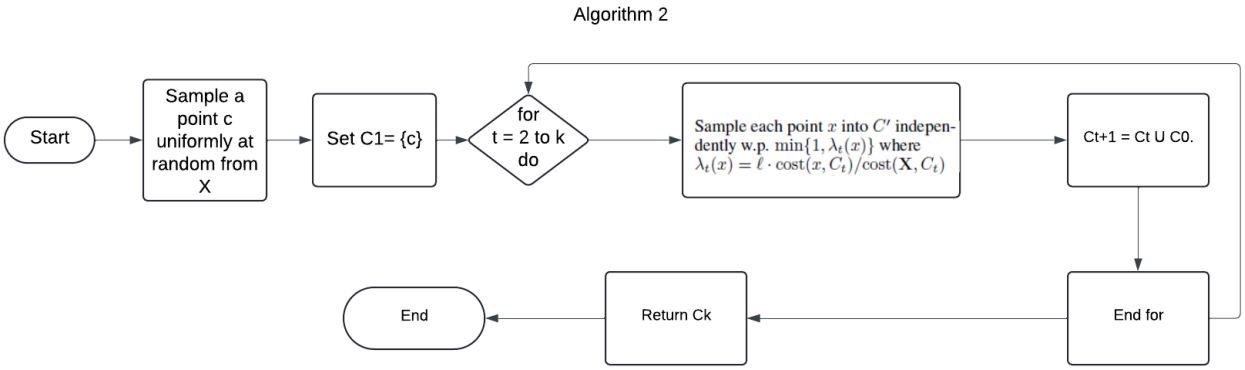
k-means++ is an enhancement over the standard k-means algorithm. It improves the initialization phase of k-means by carefully choosing the initial centroids with specific probabilities. This approach aims to minimize the potential for poor clusterings by spreading out the initial centroids before proceeding with the standard k-means algorithm.



K-means++ Parallel:

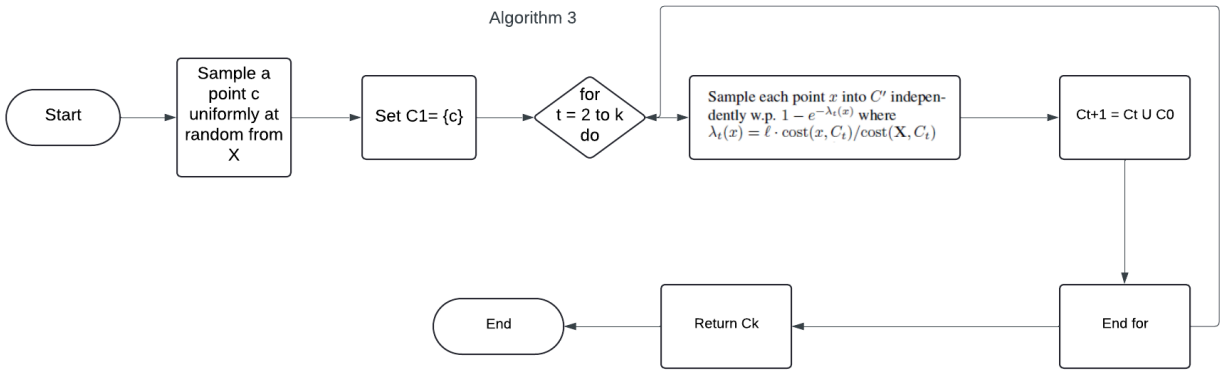
k-means|| is a parallel version of the k-means++ algorithm designed to provide a scalable solution for clustering large data sets. It initializes centroids over multiple rounds based on probability

min{ l x cost(x,C)/cost(X,C) , 1}where l is an oversampling parameter that takes values from 0.1k to 10k. T is the rounds for which the samples are sampled significantly speeding up the clustering process while maintaining an approximation to the optimal clustering solution.



K-means++ Parallel_pois Algorithm:

This is a variation of K-means++ parallel algorithm, only thing is the points are sampled with the probability of 1-exp(-l . cost(x,C)/cost(X,C)).



Key Technical Terms:

Term	**Definition**
Markov Chain Analysis	Markov Chain Analysis involves systems where the next state depends only on the current state and not on how the present state was reached (memorylessness).
D^2 Distribution	In k-means++, the D^2 Distribution is a method for selecting the next centroid in clustering. Each point in the dataset is chosen with a probability proportional to its squared distance from the nearest current centroid. This strategy is intended to improve the clustering outcome by decreasing the variance within each cluster.
Bi-criteria Approximation	Bi-criteria Approximation algorithms identify solutions by simultaneously maximizing two distinct but often opposing criteria. This method is utilized when a single best solution is not possible due to the opposing nature of the criteria.
Supermartingale Techniques	A Supermartingale is a set of random variables (usually reflecting a stochastic process) where the conditional expected value of the next variable is less than or equal to the current variable.

Summary of Main Results:

1) **Improving k-means++ Guarantees:**

The research here exceeds the existing theoretical framework by establishing that the estimated cost of the k-means++ algorithm's solution is no more than 5(ln k + 2) times the cost of the best solution. At any given stage t, the algorithm keeps a list of centers Ct. The algorithm then chooses new centers Dt+1, which may include centers that encompass any cluster Pi from the optimal clustering P. The estimated cost bound for a cluster Pi covered at stage t + 1 by centers in Dt+1 has been improved. The new bound is 5 OPT1(Pi), where OPT1(Pi) denotes the optimal cost of clustering Pi with one center. This new constraint improves upon Arthur and Vassilvitski's (2007) previous bound of 8 OPT1(Pi).

Proof of Lemma 4.1:

Setup for Lemma:

Consider a set of centers C and a subset P of dataset X. A new center c is chosen at random from P using the cost distribution for C. The analysis then determines the new cost of P by adding c to C, resulting in a new set C0.

Cost Analysis:

The cost of any point y in P after adding c to C is the lesser of the cost from y to the old set C and the distance from y to the new center c. This arrangement generates an expectation of the new cost for P based on the probability distribution of choosing any point x in P as the new center.

Symmetric Cost Function and Upper Bound:

A symmetric cost function f(x, y) is defined, which balances the cost contributions from pairs of points x and y in P. This function ensures symmetry in terms of contributions to the total cost. An upper bound on f(x, y) is then derived, showing that it is always less than or equal to 5 times the minimum of the existing cost for x and y.

Corollary 4.4, uses the supermartingale property to argue that the expected coverage cost in k-means++ never increases unexpectedly, providing a theoretical basis for robust performance guarantees in clustering applications.

2)Bi-Criteria Approximation

The efficacy of the k-means++ algorithm when additional centers, beyond the standard requirement of k, are used—a method termed as bi-criteria approximation. This modification introduces bounds that show notable improvements in the algorithm's approximation guarantees, especially when the oversampling parameter Δ is relatively small. The enhancement in performance is structured through a series of methodical steps within the proof of this theorem.

Initially, a bound on the expected cost for clusters formed after $k+\Delta$ rounds of the k-means++ algorithm. This step is premised on the assumption that clusters already covered by the initial k centers incur a minimal cost, thereby focusing the analysis on the clusters that remain uncovered after the additional Δ rounds. The subsequent step involves estimating the probability that a given cluster P remains uncovered after these rounds. This is modeled by considering a scenario in which the algorithm repeatedly selects centers from the already covered clusters without choosing any center from P, thus estimating the potential cost if P stays uncovered under the assumption that its uncovered cost remains static until it is finally covered.

Further leveraging insights from the works by Arthur and Vassilvitskii (2007) and Dasgupta (2013), the scenarios where the number of "misses" (instances where the algorithm fails to cover new clusters) exceeds $\Delta/2$. This analysis culminates in formulating two primary bounds on the expected clustering cost in relation to the optimal k-clustering cost, $\mathrm{OPT}_k(X)$. These bounds consider cases with misses fewer than $\Delta/2$ and those exceeding $\Delta/2$, with the latter offering a logarithmic approximation of the cost.

In summary, the theorem provides a sophisticated approximation guarantee for the k-means++ algorithm using $k+\Delta$ centers. By methodically calculating the impacts of additional centers beyond the standard k, the algorithm's performance across various scenarios of cluster coverage after the initial k rounds.

3)Parallel k-means++ Algorithm Analysis:

The analysis of the k-means++ parallel algorithm gives a solid theoretical basis for ensuring performance over numerous rounds. Theorem 6.1 provides upper bounds for the predicted clustering cost in two cases: when the parameter l is less than k and when l is bigger than or equal to k. The study is simplified by comparing k-means++ parallel and k-means++ parallel Pois, as well as utilizing a thought experiment with a modified version of k-meanskPois with knowledge of the best solution. They yield identical distributions between the original and updated techniques, allowing for accurate predictions of clustering behavior.

The analysis is based on Lemma 6.3, which establishes upper bounds using Poisson point processes and supermartingale theory. Applying these bounds recursively produces the total projected cost after T rounds, demonstrating that k-means++ parallel can achieve a cost that is only a minor multiple of the ideal cost. Finally, a corollary shows that after T rounds, the predicted clustering cost is no more than nine times the optimal clustering cost, assuming T is chosen based on the starting cost to optimal cost ratio. Thus, the study shows that the k-means++ parallel algorithm can provide significant cost savings with proper configuration and iteration rounds.

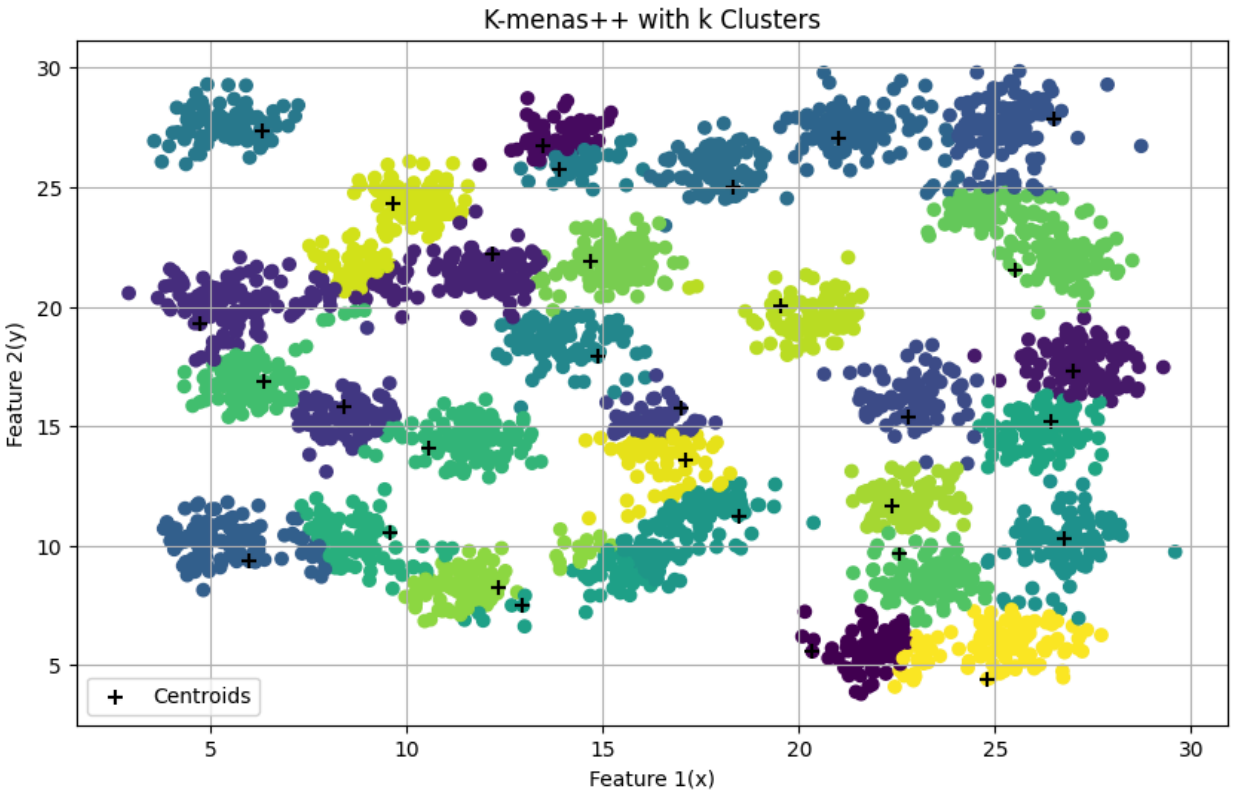
4. Application example:

Data:

The data that is used to apply the algorithms is named D31[2] that has 3100 rows of 2 dimensional data. The data is grouped into 31 clusters and this helps to find the accuracy of the algorithm that is employed on the data, as the true cluster group is known.

Application of K-means++ Clustering:

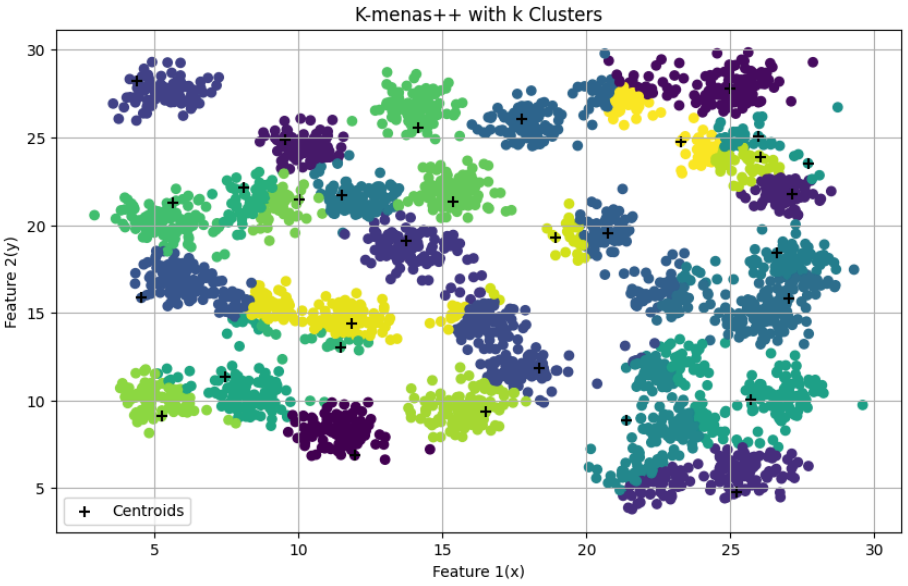
Graph:



Total Cost of the clustering: 11264.5092

Application of K-means++ Parallel Clustering:

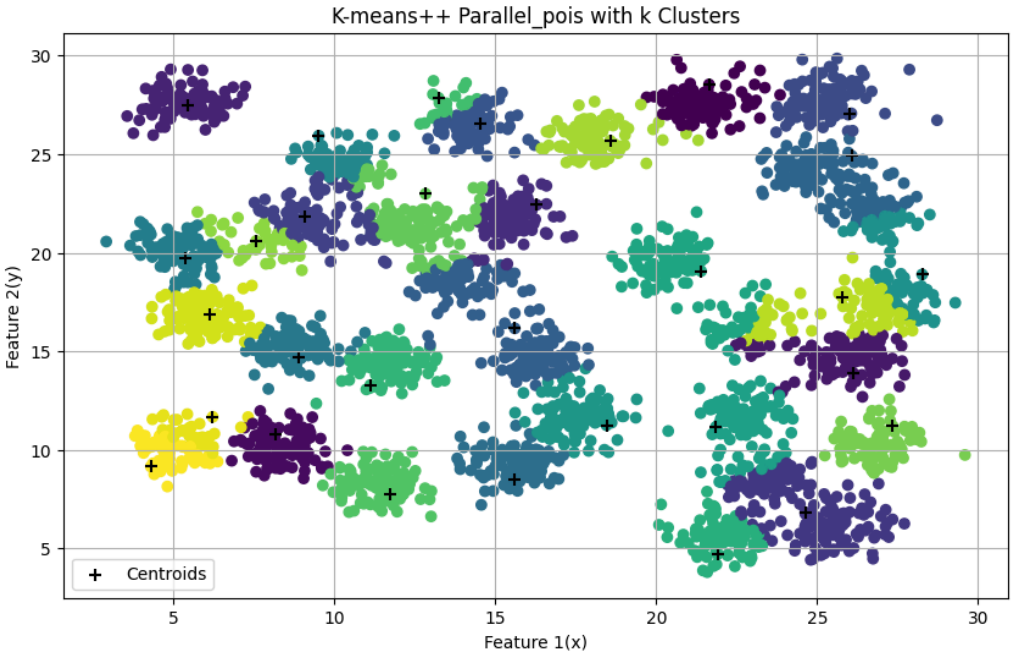
Graph:



Total Cost of the clustering: 8758.94

Application of K-means++ Parallel_pois Clustering:

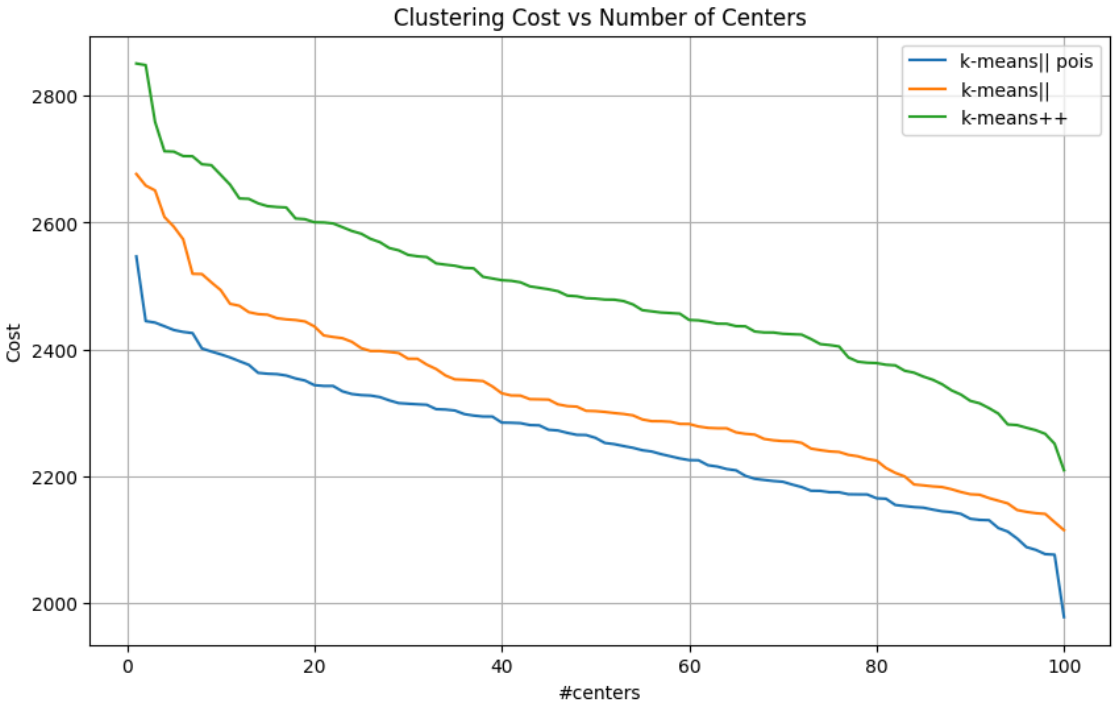
Graph:



Total Cost of the clustering: 8331.6

Experiment:

An experiment is carried out for a range of k-values from 1 to 100 on all three algorithms and the resulting graph is plotted from which we can see that k-means++ parallel_pois gives better cluster accuracy than k-means++ parallel and this gives more accuracy cluster than k-means++.



Practical verification of Improved Approximation Guarantees for k-means++:

Enhancement of Theoretical Bounds as the expected cost of the solution produced by k-means++ can now be bounded by 5(ln k + 2) times the optimal solution's cost. This improves the previous bound of 8(ln k + 2) provided by Arthur and Vassilvitskii (2007). This enhancement is achieved through a refined analysis of the expected cost of covered clusters, offering a tighter bound on these costs.

```
def simulate_kmeans_plus_plus(data_points, k, targ_clust):
    centers_sim = kmeans_plus_plus(data_points, k-1)
    new_center = kmeans_plus_plus(targ_clust, 1)[0]
    all_centers = np.vstack([centers_sim, new_center])
    return calculate_cost(data_points, all_centers), new_center
```

```
targ_clust = np.array(data[data['cluster']==10][['x','y']])
costs = []
for i in range(1000):
    cost, new_center = simulate_kmeans_plus_plus(d31_data, 31, targ_clust)
    costs.append(cost)

expected_cost = np.mean(costs)
print("Expected Cost:", expected_cost/31)
print("Optimal Cost:",optimal_cost)
optimal_cost = calculate_cost(targ_clust, [np.mean(targ_clust, axis=0)])
print("5 * OPT1(Pi):", 5 * optimal_cost)
```

```
Expected Cost: 146.03756758286096
Optimal Cost: 93.17193581863616
5 * OPT1(Pi): 465.8596790931808
```

Algorithm Complexity:

Algorithm	Seeding Phase	Time Complexity	Expected Cost
K-means ++	In the seeding phase of the k-means++ method, the initial center is selected uniformly at random, and the centers that follow are selected with a probability that is proportional to their squared distance from the nearest chosen center. Up till k centers are chosen, this center selection process is repeated.	The time complexity of the seeding step in k-means++ is mostly determined by the number of data points (n) and clusters (k). The seeding phase has a complexity of $O(nk)$O(nk), as each center selection requires calculating the distance to the nearest already taken center for all remaining points.	The k-means++ algorithm reduces the predicted approximation factor from 8(ln k + 2) to a smaller bound, resulting in greater performance guarantees.
K-means ++ Parallel	Similar to k-means++, k-means || begins by selecting the initial center	The complexity of k-means|| entails numerous passes over the	The paper offers bounds on the predicted cost of the

	at random. However, in the following rounds, several centers are selected based on a probability which is a function of the points cost. This is repeated for a set number of rounds T.	data, with each pass involving a probabilistic selection of centers using the D2 distribution. If T is the number of passes and l is a parameter defining the number of centers to choose per round, the complexity can be approximated by $O(T \cdot n \cdot l)$O(T·n·l), where each pass comprises a full scan over the dataset to update distances and select new centers.	k-means || solution, demonstrating that it has an approximation guarantee that can be improved under specific scenarios (oversampling parameters less than or greater than k).

Conclusion

In conclusion, the k-means++ and k-means++ || help enhance the existing clustering methodology by meticulously refining the initialization steps and integrating parallel processing capabilities. It improved clustering quality, faster convergence, and better scalability for large datasets over the previous existing clustering algorithms. It also integrates advanced statistical theories and methods that are applied to address the complexities of big data. These advancements directly correlate with our course, which covers a broad spectrum of statistical and computational techniques essential for big data analysis. The introduction and optimization of k-means++ align with the initial weeks of the course where fundamental clustering techniques are discussed, providing a real-world application of the theoretical concepts taught. The enhancements in the initialization phase of k-means++, which aim to minimize the total clustering cost more effectively than the traditional method, offer practical illustrations of dimension reduction and regression topics covered in subsequent weeks. The study extends the aim to optimize the clustering of big data by optimizing algorithms.

References

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

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

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

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

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

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

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

Becchetti, L., Bury, M., Cohen-Addad, V., Grandoni, F., and Schwiegelshohn, C. 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*, pp. 1039–1050, 2019.

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

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

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

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

C.J. Veenman, M.J.T. Reinders, and E. Backer, A maximum variance cluster algorithm. IEEE Trans. Pattern Analysis and Machine Intelligence 2002. 24(9): p. 1273-1280.

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

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

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

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

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

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

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

Makarychev, K., Makarychev, Y., and Razenshteyn, I. 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*, pp. 1027–1038, 2019.