A Comprehensive Survey on Centroid Selection Strategies for Distributed K-means Clustering Algorithm

Poonam Ghuli
Assistant Professor, Department of CSE,R.V. College of Engineering, Bangalore, India

Maanas Prabhakar
Department of CSE,RVCE, Bangalore, India

Rajashree Shettar
Professor & Asso. Dean, Department of CSE, R.V. College of Engineering, Bangalore, India

ABSTRACT

Extremely large data sets often known as 'Big Data' are analyzed for interesting patterns, trends, and associations, especially those relating to human behavior and interactions. Extraction of meaningful and useful information needs to be done in parallel using advanced clustering algorithms. In this paper, effort has been made to tweak in changes to the existing K-means algorithm so as to work in parallel using MapReduce paradigm. K-means due to its gradient descent nature is highly sensitive to the initial placement of the cluster centers. This random initialization of cluster centers results in empty clusters and slower convergence. In this paper, an overview of existing methods with emphasis on computational efficiency is presented. Comparison of three well known linear time complexity initialization methods has been presented here. These methods are analyzed on two different data sets. The experimental results are recorded and presented with insights on different initialization methods for practitioners.

General Terms

BigData, Unsupervised Clustering, Distributed Computing, Data Mining, Machine Learning.

Keywords

K-means Clustering Algorithm, Hadoop, MapReduce, PCA, HDFS.

1. INTRODUCTION

With the advent of modern techniques for scientific data collection, large quantities of data are getting accumulated at various databases. Systematic data analysis methods are necessary to extract useful information from this rapidly growing big data [1]. Cluster analysis is one of the major data mining methods available today. Cluster analysis seeks to partition a given data-set into groups based on specified features so that the data points within a group are more similar to each other than the points in different groups. Clustering is a crucial area of research, which finds applications in many fields, including bioinformatics, pattern recognition, image processing, marketing, data mining, and economics. Numerous methods have been proposed to solve the clustering problem. The k-means is one of the most popular clustering algorithms which is widely used for many practical applications. This paper tries to explore two important issues encountered while implementation of this popular algorithm. First, the original k-means algorithm is computationally very expensive. In addition to this, the size of modern data-sets is growing rapidly, which far exceeds the amount of memory available on even the most powerful servers. As a result, the input to massive data-set computations often cannot be stored in the memory of a single machine. To reveal the insights hidden into this huge amount of data the algorithm has to be parallelized in distributed environment. To solve these kinds of parallelizable problems involving large data-sets, the best choice is to make use of MapReduce [2-3] framework. This requires slight modification in existing algorithms to fit into MapReduce paradigm. Second, due to its gradient descent nature, it often converges to a local minimum of the criterion function. For the same reason the quality of the resulting clusters substantially relies on the choice of initial centroids. Adverse effects of improper initialization include empty clusters, slower convergence, and a higher chance of getting stuck in bad local minima. The above mentioned problems can be resolved by using adaptive initialization methods. Several methods have been proposed in the literature for improving the performance of the k-means algorithm.

This paper compares and investigates three initialization strategies which are improvement on the classic k-means algorithm to produce more accurate clusters. The three initialization methods explored are K-means with weighted average method [4], Principal component analysis [5-7] and a heuristic method [8] based on sorting and partitioning of the input data for finding better initial centroids. Experimental results show that the proposed algorithms produce better clusters in less computational time by parallelizing the tasks using Hadoop cluster setup.

The study in this paper differs from earlier studies of a similar nature [9-10] in several respects: (i) a completely different set of initialization methods are discussed and reviewed (ii) the experiments involve distributed implementation of these methods using MapReduce paradigm on a totally diverse collection of data sets, (iii) computational efficiency is used as a performance criterion, and (iv) the experimental results are analyzed more thoroughly to determine which initialization method provide better results for a given dataset. The data sets used to carry out different experiments are Temperature and Electrical dataset.

As discussed above this paper aims to demonstrate which algorithm is best suited for given dataset. In the experimental analysis it was found that K-Means Clustering using Heuristic Method and PCA gave almost similar results. These were most suited for the Year Temperature dataset. Further K-means clustering using Weighted Average is most suited for the Electrical dataset. The execution speed of Heuristic and PCA methods were found to be 9.53% and 8.85% respectively better than that of Weighted Average method for Year Temperature dataset. For the Electrical dataset, Weighted Average was found to give better execution time that was 11.11% and 4.49% faster than PCA and Heuristic method respectively.

2. RELATED WORK

Today many large-scale data processing mechanisms that have been implemented based on the original idea of the MapReduce framework are currently gaining a lot of momentum in both research and industrial communities. On top of it scalable clustering on distributed framework is considered as one of the best analysis tools for data mining applications. As a consequence, cluster analysis faces new challenges to process tremendously large and complex datasets that are stored and analyzed across large clusters of computers. To support the distributed analysis, the recent trend is to move computations (algorithms which are few KB in size) closer to data instead of moving large amount of data across machines. These algorithms process chunks of local data independently on each machine in a computing cluster. This also urged the development of new abstractions that hide system-level details from the application developer. These abstractions allow developers to concentrate on design and development of scalable algorithm that can perform large scale parallel computations without being distracted by fine grained details like concurrency management, fault tolerance, error recovery, and a host of other issues in distributed computing. For solving a problem in distributed environment the MapReduce approach is a seamless solution; however, it requires slight modification in existing algorithms to fit into MapReduce paradigm.

K-Means is the most widely used partitional clustering algorithm [11-12] which has applications in many areas such as information retrieval, computer vision, big data analytics, bioinformatics and pattern recognition to name a few. There are several reasons that make this algorithm stand out from the rest. First, it is conceptually simple and easy to implement. It's easily scalable and parallelizable. Its open source implementation is readily available in every data-mining software like WEKA, apache Mahaout (parallel implementation), scikit-learn, MS azure machine learning studio and many more. Second, it is adaptable, i.e. almost every aspect of the algorithm (initialization, distance function, termination criterion, etc.) can be modified. Third, it has a time complexity that is linear in N, D, and K (in general, D <<< N and K <<< N). Here, N represents number of data points in a data set, D is dimensionality and K is number of clusters. Fourth, it is guaranteed to converge [13] at quadratic rate [14]. Finally, it is invariant to data ordering, i.e. random shuffling of the data points. On the other hand there are many significant limitations of this popular algorithm [15]. First, k-means requires specifying k value (number of clusters) a priori and the output can vary drastically based on the number of clusters chosen. Second, Due to its gradient descent nature, it often converges to a local minimum of the criterion function. Third, presence of outliers greatly affects the means of their respective clusters due to utilization of squared Euclidean distance. This can be alleviated by using a more robust distance function. Fourth, resultant clusters formed are significantly influenced by selection of initial centroid points. Adverse effects of improper initialization include empty clusters, slower convergence and have a higher probability of getting stuck in bad local minima. All of these drawbacks except the first one can be resolved by using adaptive initialization methods.

Thus, this paper aims to compare and investigate three initialization methods in a distributed environment. This distributed implementation of initialization strategies increases the computational efficiency. Also it provides improvement on the classical k-means algorithm to produce more accurate clusters. The three initialization methods explored here are K-means with weighted average method [4], Principal component analysis [5-6] and a heuristic method [7]. The novelty in the presented work comes from the involvement of distributed implementation of initialization methods using MapReduce paradigm on a totally diverse collection of data sets. Each of the algorithms chosen has been modeled as a series of MapReduce jobs on clusters of commodity machines. Then a distributed K-Means clustering is applied onto the datasets using the carefully generated centroids. This eliminates most significant disadvantages of popular clustering algorithm. Further the experimental results are analyzed more thoroughly to determine which initialization methods provide better results for a given dataset.

In the Weighted Average algorithm, a new method is explored to find a weighted average score of dataset. In [4] Mahmud M S et al employed a uniform method to find rank score by averaging the attribute of each data point, which generated initial centroids that follow the data distribution of the given set. A sorting algorithm is applied to the computed score and divided into ‘k’ subsets, where k is the number of desired clusters. Finally, the nearest value of mean from each subset is taken as initial centroid. The initial centroids are calculated in a strategic way rather than randomly.

In their recent work [8] K A Abdul Nazeer and et al. proposed heuristic based method. The basic idea of this algorithm is to determine the initial centroids of the clusters in a heuristic manner, so as to ensure that the centroids are chosen in accordance with the distribution of data. The method involves sorting the input data set and partition the sorted data set into ‘k’ number of sets where ‘k’ is the number of clusters to be formed. Mean values of each of these sets are taken as the initial centroids. Moreover, to deal with multidimensional data they utilized an idea to determine the column with maximum range, where range is the difference between the maximum and the minimum element for each column. After identifying the attribute (column) having maximum range, the entire set of data values are then sorted in a non-decreasing order, using the Heap Sort algorithm, based on the attribute with maximum range. The sorted list of data points are then divided into ‘k’ equal sets. Finally, the arithmetic means of each of these ‘k’ sets are computed. These means become the initial centroids of the clusters to be formed. After determining the initial centroids as described above, the data points are assigned to various clusters by using the original K-means algorithm.

Principal Component Analysis [5-7] is a widely used statistical technique for unsupervised dimension reduction. It is a common technique for finding patterns in high dimensional data. The distributed PCA algorithm implemented gives the theoretical guarantee for any good approximation solution on the projected data for K-Means clustering which is a good approximation on the original data too, while the projected dimension required is independent of the original dimension [6]. The main basis of PCA-based dimension reduction is that PCA picks up the dimensions with the largest variances [7]. The first principal component is chosen as the principal axis for partitioning and sorted in ascending order. Then, dividing the set into ‘k’ subsets where k is the number of clusters. Find the median of each subset and then use the corresponding data points for each median to initialize the cluster centers.

Further, this paper presents how slight modifications in the existing algorithms to adapt to MapReduce paradigm can make many applications capable of tackling large-scale data problems. The capabilities necessary to embark upon large scale distributed data processing are already within reach by many and will continue to become more accessible over time. This large scale processing has been feasible by scaling out with clusters of commodity machines to withstand on problems of interest. By making MapReduce accessible to everyone through the open source Hadoop project had built the vibrant software ecosystem that flourishes today. Recently many improvisations are proposed of this well accepted framework in literature [16-19]. In the last decade, the MapReduce framework has emerged as a highly successful framework that has created a lot of momentum in the area of distributed computing research such that it has become the de-facto standard of big data processing platforms.

3. CENTROID INITIALIZATION MODULE

The paper explores the realization of the initial centroid selection methods for K-Means clustering algorithm on Hadoop, an open source implementation of Mapreduce paradigm. The implementation is provided for all the three initialization methods which commonly include four major modules as listed:

- Weighted Average / Heuristic / PCA Sorting Module
- Initial Centroid Selection Module
- Iterative Clustering Module
- Cluster Assignment Module

Thus, this section talk about the algorithms involved in selecting the initial centroids which provides better accuracy, before performing the K-Means Clustering. Sorting module is the first MapReduce (M/R) module to be executed in the series of three M/R jobs. This is the only module which is explicit for different initialization methods. Remaining three modules are common for the clustering process which may use one of the initialization methods. K-means clustering is implemented as a series of two M/R jobs namely Iterative Clustering Module and Cluster Assignment Module.

3.1 Sorting Module

Sorting module is required to process the dataset in such a way that it facilitates the selection of initial centroids based on any measure for selecting the central value in a given set. As mentioned above we discuss three different sorting M/R module specific for each initialization method. A generalized block diagram for the sorting module is shown below in the figure 1.

3.1.1 Weighted Average Sorting Module

The purpose of this module is to sort the data points based on the score generated by assigning weights to each attribute of the data points. This process of assigning weights enables the programmer to enhance a particular feature of the dataset, which directly affects the clustering results. A uniform rank score is assigned to each attribute by averaging over the attribute values. This module's map function is responsible for assigning weights to the attributes of a dataset, multiplies these weights with each data point, and calculates average and passes (average, datapoint) to reducer. The reducer sort the data points based upon average value and write the result to an output file. The output of this module is a text file containing the sorted data points. The output file is written to the HDFS in the predetermined folder.

3.1.2 Heuristic Sorting Module

This module reads the dataset and selects the attribute with the greatest range. The mapper then reads the selected attribute and sends the data-points to the reducer in increasing order of the attribute value. The job of the reducer is to output the sorted dataset into a text file. The output of this module is a text file containing the sorted data points. The output file is written to the HDFS in the predetermined folder.

3.1.3 Principal Component Analysis

This module reads the dataset and selects the attribute with the greatest variance. The map function then reads the selected attribute with greatest variance and sends the data-points to the reducer in increasing order of the attribute value. The job of the reducer is to output the sorted dataset into a text file. The output file containing sorted data points is written to the HDFS in the predetermined folder.

3.2 Initial Centroid Selection Module

Once the sorted data points are written to HDFS, the next step is to partition the dataset and select the k initial centroids. This is simple Java module (non MapReduce function). This module is responsible to split the dataset into k subsets. Then the median of the data points in a given subset is selected as an initial centroid, thereby obtaining k centroids that are written to an output file. This output text file is used as input to the K-means clustering algorithm along with the input data set. Block diagram for this module is as shown in Figure 2.

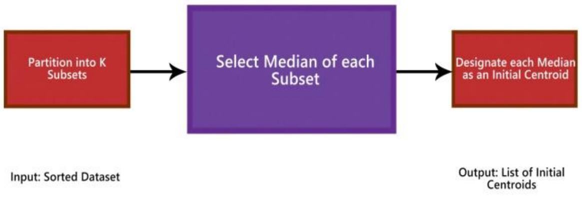
Partition into K Subsets
Select Median of each Subset
Designate each Median as an Initial Centroid
Input: Sorted Dataset
Output: List of Initial Centroids

Fig 2: Diagram for the Initial Centroid Selection Module

3.3 Iterative Clustering Module

This module is a second MapReduce job in the series that accepts a split of the dataset as input as shown in the figure 3. A setup function is used on each mapper to read the centroids and the dataset. As each data point is read, the distance between these centroids and the data point is calculated and the data point is assigned to the closet centroid. The reducer receives a pair of (K-Means centroid, list of all data points assigned in this cluster). The list is iterated to get the average data point. This is set as one of the new centroids. This is repeated for all such key, value pairs on the reducer. Finally, the centroids produced are compared to the centroids produced in the previous step to see if they have converged. If the centroids have not converged, the module is iterated with the new centroids.

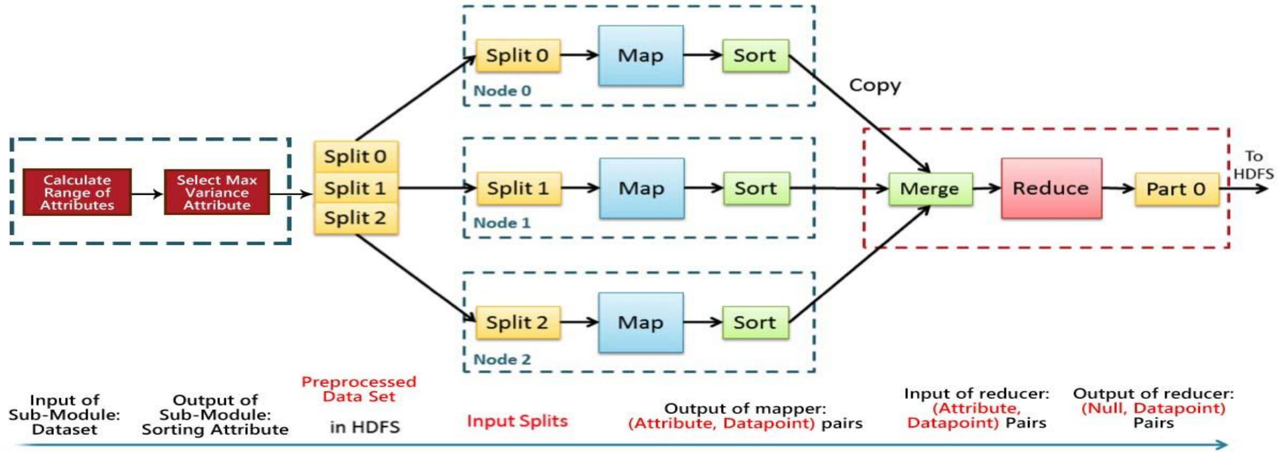
Calculate Range of Attributes
Select Max Variance Attribute
Split 0
Split 1
Split 2
Split 0
Map
Sort
Node 0
Split 1
Map
Sort
Node 1
Split 2
Map
Sort
Node 2
Copy
Merge
Reduce
Part 0
To HDFS
Input of Sub-Module: Dataset
Output of Sub-Module: Sorting Attribute
Preprocessed Data Set in HDFS
Input Splits
Output of mapper: (Attribute, Datapoint) pairs
Input of reducer: (Attribute, Datapoint) Pairs
Output of reducer: (Null, Datapoint) Pairs

Fig 1: Generalized Block Diagram for Sorting Module

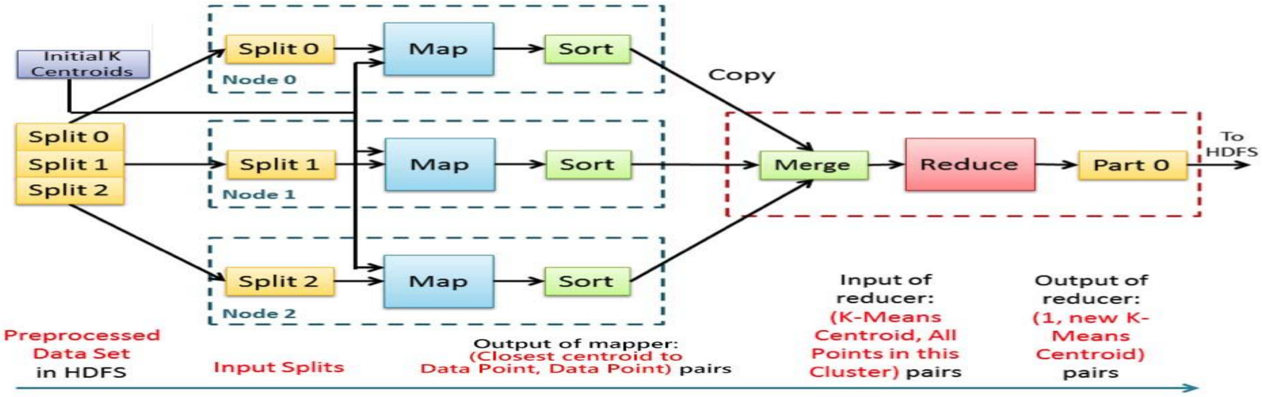
Initial K Centroids
Split 0
Split 1
Split 2
Split 0
Map
Sort
Node 0
Split 1
Map
Sort
Node 1
Split 2
Map
Sort
Node 2
Copy
Merge
Reduce
Part 0
To HDFS
Preprocessed Data Set in HDFS
Input Splits
Output of mapper: (Closest centroid to Data Point, Data Point) pairs
Input of reducer: (K-Means Centroid, All Points in this Cluster) pairs
Output of reducer: (1, new K-Means Centroid) pairs

Fig 3: Block Diagram for the Iterative Clustering Module

3.4 Cluster Assignment Module

This is the third MapReduce module in the series to implement distributed K-means clustering using initial centroid selection methods. This module accepts a split of the original data set as input and the path to the final K-Means centroid as a parameter. On each mapper, a setup function is used to read the K-Means centroids and construct a list of centroids. The distance between each data point and each K-Means centroid is calculated using a distance metric such as Euclidean distance. The data point is assigned to the cluster centroid with the least distance measure. The reducer is identity – its output is the same as its input. The final clustering file is written into the HDFS in the designated output directory.

4. EXPERIMENTAL DATASET

The implementation of the project was tested on two datasets Temperature and Electrical. The Year Temperature dataset contains 10,000 instances of different attribute values. The data set consists of two attributes, year and temperature, which specifies the average temperature for a given year. The electrical dataset consisted of around 100,000,000 instances of different attribute values. This data-set is based on recordings originating from smart plugs deployed in households. The smart plugs have sensors which are used to measure power consumption values. The value is collected roughly every second by each smart plug. The attributes of the Electrical dataset include house ID, timestamp, value (voltage), plug ID and household ID.

5. EXPERIMENTAL RESULTS AND DISCUSSION

This section presents the experimental setup for the performance evaluation of initialization methods used with k-means clustering on Hadoop framework. The algorithms were designed as a series of MapReduce jobs executed on a Hadoop cluster of 3 nodes, each with a 2.5 GHz processor and 8 GB RAM. The performance evaluated is based on time taken for the selection of the centroids by the algorithms and time taken to cluster around these centroids. Table 1 and Figure 4 analyzes the time taken in seconds by each algorithm to generate the initial centroids for K-Means Clustering with varying k values for Year Temperature dataset containing 10,000 data points. The graphs indicate that Weighted Average with 7 Clusters (k=7) is most suited for this dataset as it takes minimum time, compared to the other two algorithms.

Table 1 Analysis of Time Taken for Initial Centroids Generation (Year Temperature)

Algorithm / **K value**	**Weighted Average** (Time in Sec)	**Heuristic Method** (Time in Sec)	**Principal Component Analysis** (Time in Sec)
3 Clusters	24	26	26
5 Clusters	24	27	27
7 Clusters	23	27	27

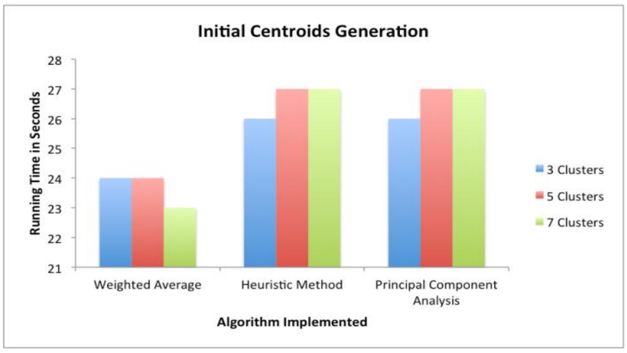

Fig 4: Analysis of Time Taken for Initial Centroids Generation (Year Temperature)

Table 2 and Figure 5 analyze the time taken by each algorithm to generate the initial centroids for K-Means Clustering with varying K values for Electrical dataset. It indicates that Weighted Average with 3 Clusters (k=3) and 7 Clusters (k=7) is most suited for this dataset as it is faster than other two methods.

Table 2 Analysis of Time Taken for Initial Centroids Generation (Electrical)

Algorithm / K value	Weighted Average (Time in Sec)	Heuristic Method (Time in Sec)	Principal Component Analysis (Time in Sec)
3 Clusters	22	25	26
5 Clusters	23	26	34
7 Clusters	22	25	26

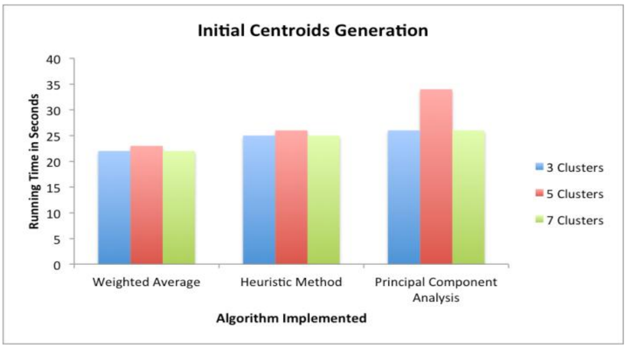

Fig 5: Analysis of Time Taken for Initial Centroids Generation (Electrical)

The previous figures indicate the time taken to find initial centroids in a systematic way. Next, the given dataset must be clustered using these generated centroids. Thus, Table 3 and Figure 6 analyze the time taken by each algorithm to perform the clustering part of the algorithm on Year Temperature dataset. It must be noted here that the time taken by simple K-Means Clustering varies based on the random selection of initial centroids and hence shouldn't be considered. The tabulated values in Table 3 demonstrate that K-Means Clustering using Heuristic Method for 5 Clusters (k=5) is most suited.

Table 3 Analysis of Time Taken for Clustering (Year Temperature)

Algorithm	K-Means Clustering (sec)	Weighted Average (sec)	Heuristic Method (sec)	Principal Componen t Analysis (sec)
3 Clusters	67	123	107	108
5 Clusters	52	94	65	66
7 Clusters	78	88	79	77

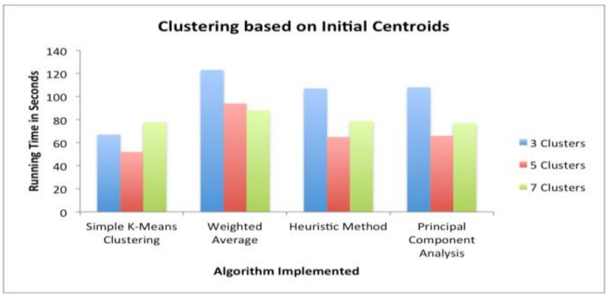

Fig 6: Analysis of Time Taken for Clustering (Year Temperature)

Table 4 and Figure 7 analyze the time taken by each algorithm to perform Clustering part for Electrical dataset. It indicates that K-Means Clustering using Weighted Average for 3 Clusters (k=3) and K-Means Clustering using Heuristic Method for 5 Clusters (k=5) perform identically and are most suited for this dataset as they perform clustering faster than weighted average method.

Table 4 Analysis of Time Taken for Clustering (Electrical)

Algorithm / K-value	K-Means Clustering (sec)	Weighted Average (sec)	Heuristic Method (sec)	Principal Component Analysis (sec)
3 Clusters	86	63	64	64
5 Clusters	97	65	63	65
7 Clusters	95	66	64	64

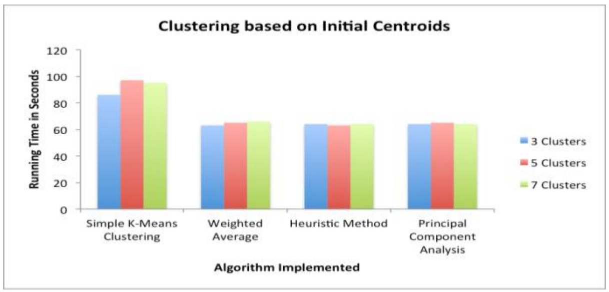

Fig 7: Analysis of Time Taken for Clustering (Electrical)

Table 5 and Figure 8 analyze the total time taken to perform Initial Centroid Generation and k-means Clustering for each algorithm on Year Temperature dataset. It indicates that K-Means Clustering using Heuristic Method for 5 Clusters (k=5) is most suited for this dataset as indicated in Table 5.

Table 5 Analysis of Total Time taken including Initial Centroid Generation and Clustering (Year Temperature)

Algorithm K-value	K-Means Clustering (sec)	Weighted Average (sec)	Heuristic Method (sec)	Principal Component Analysis (sec)
3 Clusters	67	147	133	134
5 Clusters	52	118	92	93
7 Clusters	78	111	106	104

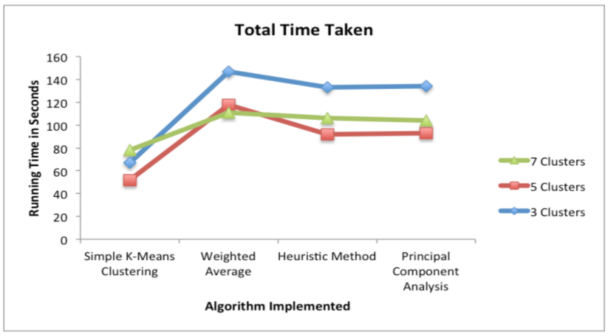

Fig: 8 Analysis of Total Time Taken taken including Initial Centroid Generation and Clustering (Year Temperature)

Table 6 and Figure 9 analyze the total time taken to perform Initial Centroid Generation and k-means clustering for each algorithm on Electrical dataset. It indicates that K-Means clustering using Weighted Average for 3 Clusters (k=3) is most suited for this dataset.

Table 6 Analysis of Total Time taken including Initial Centroid Generation and Clustering (Electrical)

Algorithm K-value	K-Means Clustering (sec)	Weighted Average (sec)	Heuristic Method (sec)	Principal Component Analysis (sec)
3 Clusters	85	85	89	90
5 Clusters	97	88	89	99
7 Clusters	95	88	89	90

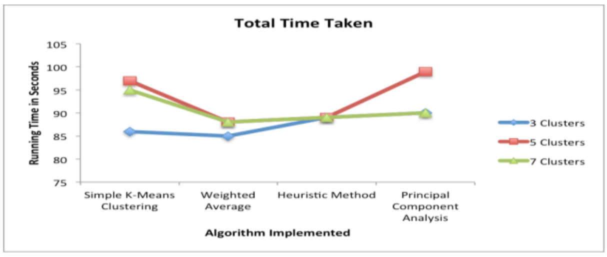

Fig 9: Analysis of Total Time Taken including Initial Centroid Generation and Clustering (Electrical)

Visualization is a crucial component of data mining. Scatterplots is the visualization tool used for presenting the clustered results. These visualizations also assist to open up some facts and observations about the underlying data which may not be possible from statistical analysis.

Figure 10 shows the clusters generated by K-Means clustering using Weighted average for 5 Clusters (k=5) on Year Temperature dataset. The clusters are not evenly spaced out as the centroid selection is dependent on the sorted average score.

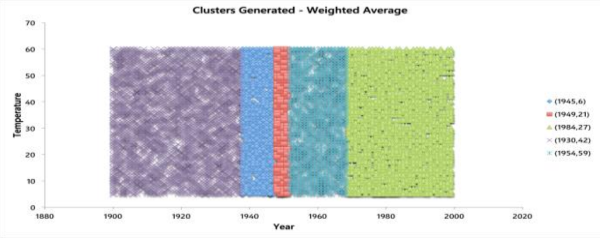

Fig 10: Clusters Generated (Year Temperature): Weighted Average

Figure 11 shows the clusters generated by K-Means Clustering using Heuristic Method for 5 Clusters (k=5) on Year Temperature dataset. The clusters are evenly spaced out as the centroids have been selected after sorting the dataset based on the range of the attributes.

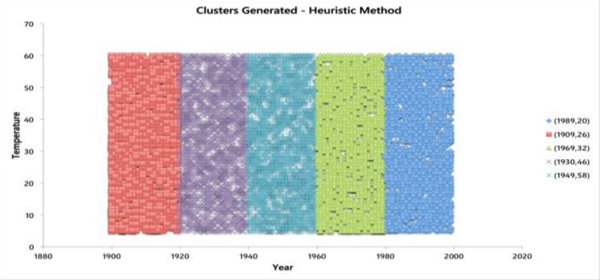

Fig 11: Clusters Generated (Year Temperature): Heuristic Method

The experimental analysis reveals that K-Means Clustering using Heuristic Method with 5 Clusters (k=5) is most suited for the Year Temperature dataset. The experimental analysis also reveals that K-Means Clustering using Weighted Average with 3 Clusters (k=3) is most suited for the Electrical dataset. Since, K-Means Clustering randomly chooses centroids, hence the time required for clustering is not constant each time it is executed. Thus, it may not be fair to consider the execution of this random clustering algorithm. The analyses disclose that K-Means clustering using Heuristic Method and Principal Component Analysis give similar performance for a given number of clusters.

6. CONCLUSION AND FUTURE WORK

This paper presents a comprehensive review of different initialization methods for carefully selecting centroids for k-means clustering in a distributed environment using MapReduce framework. Further, the experiments conducted are analyzed more thoroughly to determine which initialization method provides better results for a given dataset. The computational efficiency is used as the performance criteria for selecting appropriate initialization method for a given dataset.

For the Temperature dataset, Heuristic method and PCA have shown execution speeds of 9.53% and 8.85% respectively, better than that of Weighted Average method. Performance of K-means for weighted average method with respect to temperature dataset creates an overhead when weights are added to attributes resulting in increased clustering time and also the centroids are not evenly spaced. Thus, making it unsuitable for clustering on such dataset.

K-Means Clustering using Heuristic Method and Principal Component Analysis considers all dimensions of a given data set and thus selects the best possible attribute that is employed to choose initial centroids. This results in selection of evenly spaced centroids. Hence, these two algorithms give similar performance on the temperature dataset.

It was found that for the Electrical dataset, Weighted Average was found to give improved overall execution time that is 11.11% and 4.49% faster than PCA and Heuristic method respectively. This happens due to the fact that experiments were conducted on relatively small sample of the Electrical dataset and the computational time involved in working with large timestamp values.

The results obtained from the experiments clearly show that the process of clustering depends upon several factors. These factors include the type of data, number of dimensions, the size of the dataset, the attributes on which clustering is performed and the number of clusters chosen ('k').

Future work can be focused on trying different initialization methods on more diverse collection of datasets to gain further insights into the data. One can even use methods to automate the process of determining appropriate value of k based on input data. For example Silhouette Coefficient can be used for this. An attempt can be made to evaluate the performance of clustering using internal and external indexes. External index is used to measure the extent to which resultant cluster labels match to the externally supplied ground truth set of classes. This includes Adjusted Rand Index, V-Measure, Mutual Information based scores etc. Internal index is used to measure the goodness of a clustering structure without the requirement of external information or truth values. Internal index includes score based on Silhouette Coefficient value. Further, iterative MapReduce frameworks like Twister [19] can be used to yield better performance boost for iterative algorithms.

7. REFERENCES

[1] Min Chen, Shiwen Mao, Yunhao Liu, "Big Data: A Survey", Mobile Networks and Applications, Springer publication, Volume 19, (2), April 2014, pp 171-209

[2] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM, Vol 51, no. 1 (2008): 107-113.

[3] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters." In Sixth Symposium on Operating System Design and Implementation (OSDI 2004), Dec 2004, pp. 137-150.

[4] Mahmud, M. S., Rahman, M. M., & Akhtar, M. N. (2012, December). Improvement of K-means clustering algorithm with better initial centroids based on weighted average. In *Electrical & Computer Engineering (ICECE), 2012 7th International Conference on* (pp. 647-650). IEEE.

[5] Ding, C., & He, X. (2004, July). K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning* (p. 29). ACM.

[6] Ding, C. H., & He, X. (2004, April). Principal Component Analysis and Effective K-Means Clustering. In *SDM* (pp. 497-501).

[7] D.Napoleon, S.Pavalakodi "A New Method for Dimensionality Reduction using KMeans Clustering Algorithm for High Dimensional Data Set" International Journal of Computer Applications , Vol. 13, (7), January 2011

[8] Nazeer, K., Kumar, S. M., & Sebastian, M. P. (2011, February). Enhancing the k-means clustering algorithm by using a O (n logn) heuristic method for finding better initial centroids. In *Emerging Applications of Information Technology (EAIT), 2011 Second International Conference on* (pp. 261-264). IEEE.

[9] J. M. Pena, J. A. Lozano, P. Larranaga, 1999, An Empirical Comparison of Four Initialization Methods for the K-Means Algorithm, Pattern Recognition Letters 20 (10) (1999) 1027–1040.

[10] J. He, M. Lan, C. L. Tan, S. Y. Sung, H. B. Low 2004 Initialization of Cluster Refinement Algorithms: A Review and Comparative Study, in: Proc. of the 2004 IEEE Int. Joint Conf. on Neural Networks, pp. 297–302.

[11] A. K. Jain, M. N. Murty, P. J. Flynn 1999 Data Clustering: A Review, ACM Computing Surveys, 31 (3), pp 264–323.

[12] A. K. Jain 2010 Data Clustering: 50 Years Beyond K-means, Pattern Recognition Letters 31 (8) 651–666.

[13] S. Z. Selim, M. A. Ismail, K-Means-Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality, IEEE Trans. on Pattern Analysis and Machine Intelligence 6 (1) (1984) 81–87.

[14] L. Bottou, Y. Bengio, Advances in Neural Information Processing Systems 7, MIT Press, 1995, Ch. Convergence Properties of the K-Means Algorithms, pp. 585–592.

[15] Celebi, M. Emre, Hassan A. Kingravi, and Patricio A. Vela. "A comparative study of efficient initialization methods for the k-means clustering algorithm." Elsevier, Expert Systems with Applications, 40, no. 1 (2013): 200-210.

[16] E. Dede, Z. Fadika, M. Govindaraju, and L. Ramakrishnan, "Benchmarking MapReduce implementations under different application scenarios," in Future Generation Computer Systems, Special Section: eScience Infrastructure and Applications, Vol. 36. IEEE Computer Society Washington, DC: Elsevier, 2014, pp. 389-399.

[17] S. Sakr, A. Liu, and A. G. Fayoumi. "The family of MapReduce and large-scale data processing systems," ACM Comput. Surv. (CSUR), Vol. 46, no. 1, Oct. 2013, Article no. 11.

[18] M. Yoon, H.-il Kim, D. H. Choi, H. Jo, and J.-w. Chang, "Performance analysis of MapReduce-based distributed systems for iterative data processing applications," Mobile Ubiquit. Intell. Comput., Vol. 274, no. 1, pp. 293_299, Mar. 2014.

[19] Ghuli, P., Shukla, A., Kiran, R., Jason, S., & Shettar, R. (2015). Multidimensional Canopy Clustering on Iterative MapReduce Framework Using Elefig Tool. IETE Journal of Research, 61(1), 14-21.