KM-MBFO: A Hybrid Hadoop Map Reduce Access for Clustering Big Data by Adopting Modified Bacterial Foraging Optimization Algorithm

Suja C Nair, M. Sudheep Elayidom, Sasi Gopalan

Abstract--- K-Means Clustering is a very powerful and frequently used algorithm for the clustering, it has got its own limitation. The prevalent K-Means clustering algorithm used for grouping have inadequacies, for example, slow convergence rate, local optima trap, and so on. Therefore, many swarm knowledge based procedures combined with KM for clustering were presented and demonstrated their presentation, its variations and its applications in data grouping. In this paper we intend to propose a parallel organizing strategy for KM-MBFO mechanism that actualized in Hadoop Distributed File System (HDFS) for diminishing the execution time. This Mapper approach produces the populace for given data set for grouping. The Modified Bacterial Foraging Optimization (MBFO) algorithm finds the wellness of the populace to choose the optimal K values as far as execution time and classification error. Through simulated test results, we assess the demonstration of the proposed KM-BFO conspire.

Keywords--- HDFS, Modified Bacterial Foraging Optimization (MBFO), K-Means Clustering, Big Data.

I. INTRODUCTION

With the advancement of technology, the amount of data that is being created and stored on a global level is almost inconceivable and keeps growing. Data volumes processed by many applications crosses the peta-scale threshold and this massive volume of data is termed as big data [1]. It is estimated that the data is growing at a 40% compound annual rate, reaching nearly 45 Zeta bytes by 2020. Big data is an assortment of so large and complex that it becomes difficult to process using conventional database management tools. The challenges include capture, storage, search, sharing, analysis, and visualization. The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets [2]. Data-driven decision making as well as the burgeoning demand for data analytics has inspired increasing numbers of scholars as well practitioners to develop and apply clustering algorithms [3].

Generally, there are four categories of clustering methods: Hierarchical method is based on the distance between objects and clusters. The idea of hierarchical methods is that objects are more related to nearby objects rather than the farther objects. A balanced iterative reducing clustering hierarchy (BIRCH) [4] is the well-known algorithm in this category. The second category is the partitioning method, the main idea is that it construct k (k<n) partitions and then evaluate them by some criterion, for example, minimizing the sum of square errors. Typical algorithms include k-means and affinity propagation clustering (AP) [5]. Density-based method is the third category, clusters are dense regions in the data space, separated by regions of lower object density and a cluster is defined as a maximal set of density-connected points. The fourth is the model-based method, which is hypothesized for each of the clusters and tries to find the best fit of that model to each other; the well-known algorithm is expectation maximization clustering (EM) [6]. However, the conventional data analytics are becoming less suitable for handling the current big data processing [7].

To meet the challenge of big data applications, researchers proposed new methods and extended standard clustering algorithms to tackle big data which is too large to load it all into the memory for deriving clusters. K-means [8] was extended to hybridize with particle swarm algorithm k-means clustering (CPSO) [9] for enhancing the search ability over high dimensional data by Tang in 2012.Data clustering can be easily cast as a global optimization problem of finding the partition that maximizes/minimizes an objective function [10]. This can be appropriately tackled using meta-heuristics, such as Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), Particle Swarm Optimization (PSO) and so forth. These meta-heuristics exhibit different dynamics leading to distinct strategies that can be effectively combined to handle hard optimization problems such as Clustering large data sets

II. RELATED WORKS

Numerous research works have previously existed in literature which was based on the big data based clustering techniques and schemes. Some of the works are reviewed here

Rui Tang and Simon et al.[11] presented a new partitioned clustering method that is optimized by meta-heuristic for IoT big data environment. Their method has three main activities: Firstly, a sample of the data set was partitioned into mini batches. It was followed by adjusting the centroids of the mini batches of data.

Manuscript received September 16, 2019.

Suja C Nair, Research Scholar, Cusat, Cochin. India. (e-mail: nairsuja34@gmail.com)

M. Sudheep Elayidom, Professor, School of Engineering, Cusat, Cochin. India (e-mail: sudheepelayidom@gmail.com)

Sasi Gopalan, Professor, School of Engineering, Cusat, Cochin. India .(e-mail: sgcusat@gmail.com)

The third step was collating the mini batches to form clusters, so the quality of the clusters would be maximized. How the positions of the centroids could be optimally attuned at the mini batches
were governed by a meta-heuristic called Dynamic Group Optimization. The data were processed in parallel in Hadoop. Extensive experiments were conducted to investigate the performance. The results showed that their method was a promising tool for clustering fused IoT data efficiently.

Zakaria Benmounah et al.[12] developed a decentralized distributed big data clustering solution using three swarm intelligence algorithms according to MapReduce framework. The developed framework was based on cooperation between the three algorithms namely Particle Swarm Optimization, Ant Colony Optimization and Artificial Bees Colony to achieve largely scalable data partitioning through a migration strategy. That latter reaps advantage of the combined exploration and exploitation capabilities of these algorithms to foster diversity. The framework was tested using Amazon Elastic MapReduce service (EMR) deploying up to 192 computer nodes and 30 gigabytes of data. Parallel metrics such as speed-up, size-up and scale-up are used to measure the elasticity and scalability of the framework. The results were compared with their counterpart's big data clustering results and showed a significant improvement in terms of time and convergence to good quality solution. The developed model has been applied to epigenetics data clustering according to methylation features in CpG islands, gene body, and gene promoter in order to study the epigenetics impact on aging. Experimental results revealed that DNA-methylation changes slightly and not aberrantly with aging corroborating previous studies

Dheeraj Kumar et al. [13] developed a new clusiVAT algorithm and compare it with four other popular data clustering algorithms. Three of the four comparison methods were based on the well-known, classical batch k-means model. clusiVAT was based on sampling the data, imaging the reordered distance matrix to estimate the number of clusters in the data visually, clustering the samples using a relative of single linkage (SL), and then noniteratively extending the labels to the rest of the data-set using the nearest prototype rule. They have performed experiments to show that k-means and its modified algorithms suffer from initialization issues that cause many failures. On the other hand, clusiVAT needs no initialization, and almost always finds partitions that accurately match ground truth labels in labeled data. CURE also finds SL type partitions but was much slower than the other four algorithms. In their experiments, clusiVAT proves to be the fastest and most accurate of the five algorithms.

Chowdam Sreedhar et al. [14] presented two approaches to the clustering of large data sets using MapReduce. The first approach, K-Means Hadoop MapReduce (KM-HMR), focuses on the MapReduce implementation of standard K-means. The second approach enhances the quality of clusters to produce clusters with maximum intra-cluster and minimum inter-cluster distances for large data sets. The results of the proposed approaches showed significant improvements in the efficiency of clustering in terms of execution times. Experiments conducted on standard K-means and proposed solutions showed that the KM-I2C approach was both effective and efficient.

Minyar Sassi Hidri et al. [15] in their paper, made full use of consensus clustering to handle Big Data clustering. They used sampling combined with a split-and-merge strategy to fragment data into small subsets, then basic partitions are locally generated from them using parallel processing MapReduce model and later a consensus tendency was followed to obtain the final result. A scalability analysis was conducted to demonstrate the performance of their proposed clustering models by increasing both the number of computing nodes used and the sample size while satisfying the volume and the velocity dimensions.

N. Bharill et al. [16] have concentrated on the structure and execution of partitional put together grouping calculations with respect to Apache Spark, which were appropriate for bunching extensive data sets because of their low computational prerequisites. They propose a Scalable Random Sampling with Iterative Optimization Fuzzy c-Means calculation (SRSIO-FCM) executed on an Apache Spark Cluster to deal with the difficulties related with enormous information grouping. Exploratory investigations on different enormous data sets have been directed. The exhibition of SRSIO-FCM was made a decision in correlation with the proposed adaptable variant of the Literal Fuzzy c-Means (LFCM) and Random Sampling in addition to Extension Fuzzy c-Means (rseFCM) actualized on the Apache Spark group. The relative outcomes are accounted for as far as reality intricacy, run time and proportion of bunching quality, demonstrating that SRSIO-FCM can keep running in substantially less time without trading off the grouping quality.

K. Peng et al. [17] has proposed a clustering strategy for IDS dependent on Mini Batch K-means joined with important part examination. Initial, a pre-processing technique was proposed to digitize the strings and afterward the informational index was standardized in order to improve the clustering proficiency. Second, the essential part investigation technique was utilized to diminish the component of the prepared informational collection planning to additionally improve the clustering productivity, and afterward little clump K-means strategy was utilized for information grouping. All the more explicitly, we use K-means++ to instate the focuses of group so as to maintain a strategic distance from the calculation getting into the nearby ideal, also, they pick the Calsski Harabasz pointer with the goal that the clustering result was all the more effectively decided. Contrasted and different strategies the trial results and the time multifaceted nature examination demonstrate that our proposed technique was compelling and proficient. Most importantly, our proposed grouping technique can be utilized for IDS over enormous information condition.

J. Li and H. Liu et al. [18] have envisioned the challenges for big data analytics.

To facilitate and promote feature selection research, they present an open source feature selection repository (scikit-feature) of popular algorithms. We're surrounded by huge amounts of large-scale high-dimensional data, but learning tasks require reduced data dimensionality. Feature selection has shown its effectiveness in many applications by building simpler and more comprehensive models, improving learning performance, and preparing clean, understandable data. Some unique characteristics of big data such as data velocity and data variety have presented challenges to the feature selection problem.

III. PROPOSED METHODOLOGY

Today is the world of computer, technologies and computing devices, because new technologies have made computer systems faster and more affordable. Due to the growing number of labs, departments, and institutions high-performance parallel systems are required, such as clusters. Because a single computing system has limited computing resources, so to satisfy the increasing computational demands there is one method to utilize computing resources across multiple distributed computing systems. Here a Hybrid Approach for Clustering Big Data by Adopting Modified Bacterial Foraging Optimization Algorithm is proposed for parallel processing.

- We employ a Bacterial Foraging Optimization Algorithm (BFO) to tackle the local optimality problem in K-Means clustering technique and to produce tighter and more cohesive clusters based on incorporating criteria for these characteristics in the objective function.
- We design parallel processing scheme for KM-BFO mechanisms that implemented in Hadoop environment for reducing the execution time.
- Mapper approach generates the population for given data set for clustering. The reducer approach finds the fitness of the population to select the optimal clusters in terms of execution time and classification error.
- Through extensive experimental results, we evaluate the performance of the proposed KM-BFO scheme.

3.1. Preliminaries

K-means Clustering Algorithm [19]

K-Means is a standout amongest the most well-known and powerful segment based grouping techniques. It works in an iterative way to allot data points to the cluster centers which are picked chaotically, and toward the finish of every emphasis the process update the cluster centers. The best number of group's k prompting the best detachment (separate) isn't known as from the earlier and must be enumerated from the data. The target of K-Means clustering is to limit complete intra-cluster fluctuation, or, the squared error effort: The procedure proceeds for a fixed number of cycles until there is no adjustment in the center points.

Step 1: The big data in Clustered into k groups where k is predefined.

Step 2: The cluster center is selected randomly which is represented as 'Ci'.

Step 3: Based on Euclidean distance function, the objects are assigned to their closer cluster centers

Step 4: For all the objects in the each cluster group calculate the centroid or mean

Step 5: In sequential rounds repeat steps 2, 3 and 4 until the same points are assigned to each cluster

Modified Bacterial Foraging Optimization (MBFO) Algorithm [20]

BFO calculation is motivated by the searching and chemotaxis conduct of E. coli bacterium. The E. coli bacterium is one of the old diversity bacterium which has been investigated. It has a plasma layer, cell divider, and case that contain the cytoplasm and nucleoid. In addition, it has a few flagella utilized for movement. On the off chance that the flagella turn counter clockwise, they will push the cell thus the bacterium keeps running towards. Also, if the flagella pivot clockwise, they will pull the cell and the bacterium tumbles [3]. With running and tumbling, microorganisms endeavor to discover nourishment and keep away from hurtful marvels, and the microscopic organisms appear to purposefully move as a gathering. By reenacting the searching procedure of microorganisms, Passino proposed the BFO calculation. The fundamental paces of BFO are clarified as pursued.

Chemotaxis

As it described above, a chemotactic step is define to be a tumble followed by a tumble or a tumble followed by a run. The formula of position change of a bacterium is given as Eq. (1).

$$\theta_i^{t+1} = \theta + C(i)\phi(i) \quad (1)$$

Where,

θ_i^t - Position of ith bacterium in the tth chemotaxis step

$C(i)$ - Chemotactic step size

$\phi(i)$ - tumble angle

In each chemotactic step, the bacterium created an arbitrary tumble bearing initially. At that point the bacterium moves toward the path. On the off chance that the nutrient concentration in the new position is superior to the last position, the bacterium will run one more advance in the similar way. On the off chance that the nutrient deteriorate or the most extreme run step is achieved, it will complete the run and sit tight for the following tumble. The greatest run times in the calculation is constrained by a parameter called N_s

Swarming

At the point when a lot of E. coli cells are set amidst a semisolid agar with particular nutrient chemo-effectors, they move far from the center in a voyaging circle of cells by climbing the nutrient angle delivered by use of the nutrient by the set. To achieve this, reproduce the cell to cell signaling through an attractant and a repellant.

Reproduction

For each N_c time of chemotactic steps, a development step is taken in the microscopic organism's populace. The bacteria populace is arranged in descending order by their nutrient esteems. The reproduction step can be represented by equation (2)

$$J_{health}^{i} = \sum_{j=1}^{N_c+1} J(i, j, k, l) \quad (2)$$

Where,

J_{health}^{i} - Health of bacterium

At that point the least healthy microorganisms' dies and the other most beneficial microbes each split into two microscopic bacterial organisms, which are set at a similar area. For the most part, rises to half of the populace size to keep the populace estimate as steady, which is advantageous in coding the calculation. Using reproduction operator, bacteria with higher nutrients are endure and repeated, which ensures the potential optimal regions are looked all the more cautiously.

Eliminate and Dispersal

This procedure is recognized as elimination and dispersal in the developmental technique. After each multiple times of reproduction steps, an eliminate-dispersal occasion occurs. For every bacterium, it will be scattered to an irregular area as indicated by certain likelihood. This operator could improve the decent variety of the algorithms.The microorganisms are disposed of in BFO calculation, with the probability of Ped. On the off chance that a bacterium is dispensed with, simply scatter one to an arbitrary position on the enhancement territory so as to keep up the quantity of bacterium in the steady populace. Despite the fact that elimination and dispersal pulverizes the chemotactic advance, it likewise perhaps aids chemotactic, since dispersal may put the microscopic organisms close to food sources. Subsequently from a wide view, it tends to be said that elimination and dispersal is a piece of populace level versatile conduct.

Here if Ped is large, the calculation can corrupt to irregular thorough the search process. Thus just if Ped is picked properly, the calculation can be made to hop from nearby optima to worldwide optima. Since, the assessed estimation of Ped can be picked just arbitrarily and the procedure progresses toward becoming a tedious process. Therefore in order to overcome the above problems in this paper we have used a modified BFO algorithm in the elimination and dispersal step which is a less time consuming procedure.

3.2. Parallel Processing Scheme for KM-MBFO Mechanisms in Hadoop

In this paper we intend to propose a Hadoop MapReduce hybrid K-means clustering algorithm and a Modified bacterial foraging Optimization (MBFO) algorithm to manage vast data set. The mean value estimations of data indicated in clusters are utilized to measure cluster the quality of being similar to something.

Then Map Reduce approach is used to calculate the distance between the information focused simultaneously. The main advantage of this algorithm is its less mathematical complexity. As we are using the map reduce approach fast convergence is possible. Accuracy and wide application is another advantage of this algorithm. MBFO is not largely affected by the size and non- linearity of the problem.

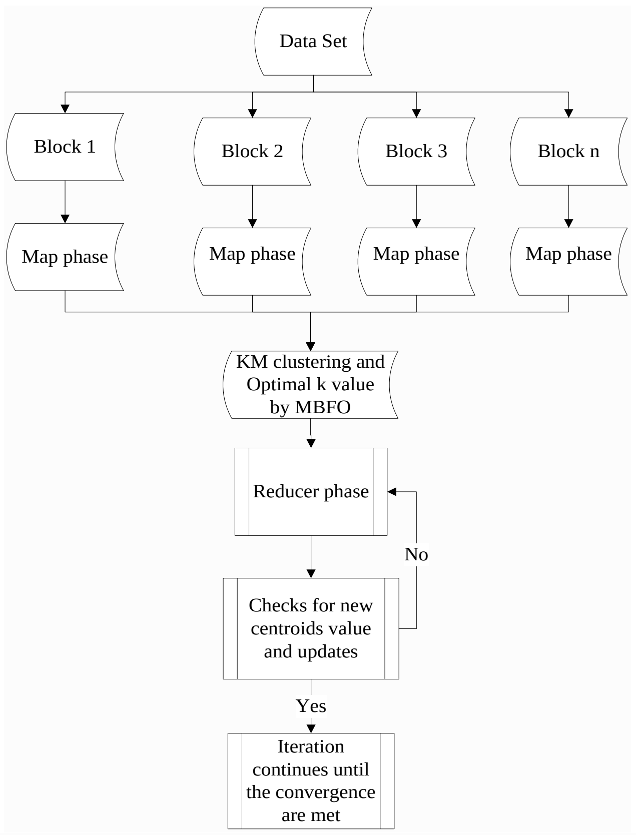
Data Set
Block 1
Block 2
Block 3
Block n
Map phase
Map phase
Map phase
Map phase
KM clustering and Optimal k value by MBFO
Reducer phase
No
Checks for new centroids value and updates
Yes
Iteration continues until the convergence are met

Figure 1: Parallel processing scheme of Proposed KM-MBFO

When grouping expansive data sets, parallel handling is most appropriate to Hadoop MapReduce because of its ability to separate extensive data sets into lumps and to store them in laborer nodes. The hybrid KM-MBFO calculation is adequate to expansive data sets however for a decent clustering algorithm tending to substantial data sets, cluster efficiency and Euclidean distance based cluster measures ought to be viewed as separated from execution times and ought to be scaled up to meet input data estimate prerequisites.

For large data set, the processing and result generation process takes a considerable amount of time and our proposed hybrid KM-MBFO approach speeds up the process within a reasonable execution time, the proposed parallel processing scheme is depicted in figure 1.

Steps Involved in our Proposed Approach

- The map phase stores input data sets in a HDFS which is utilized to relegate information objects or indicates the nearest focus and which decreases works in updating determined centers dependent on the Euclidean distance.

- KM-MBFO focuses on the MapReduce implementation of standard K-means, which serves as a framework for parallelization problems (e.g., clustering) which involves two major phases: a map phase and a reduce phase.
- The MapReduce job involves splitting a data set into parts of a fixed sized referred to as chunks.
- The map phase figures the separations between each item and each group and allots each object to its closest cluster. One map task is made for each input split and is executed by map abilities for each record of the data/information split.
- Initial K data points as cluster centers were selected optimally by means of MBFO algorithm and update the centroid value for each and every iteration
- In the map phase the data file with cluster centers form a key value and the distance between the clusters were computed by means of Euclidean distance.
- The objects among similar cluster are sent to reducer stage. The reducer stage computes the new group by combining and finds centroids for the following MapReduce work.
- In order to improve the clustering process efficiency, we make use of an MBFO algorithm and the large data sets described in Data sets.
- Figure 2 portrays the general progression of KM-MBFO. Group centroids delivered toward the end of an underlying cycle are put away in an old cluster record and are verified for the presence of new cluster centroids with every iteration process.
- In mapping phase, assigning the data points to the nearest clusters. In the combine phase clustered data is gone through optimization for the centroids of the clusters. Then check it out with the old one and update.

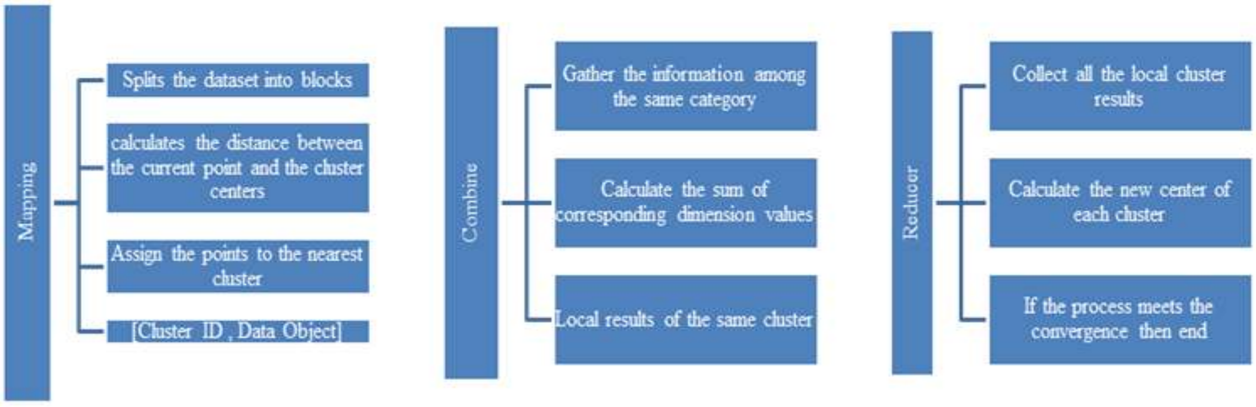
Mapping
Splits the dataset into blocks
calculates the distance between the current point and the cluster centers
Assign the points to the nearest cluster
[Cluster ID , Data Object]
Combine
Gather the information among the same category
Calculate the sum of corresponding dimension values
Local results of the same cluster
Reducer
Collect all the local cluster results
Calculate the new center of each cluster
If the process meets the convergence then end

Figure 2: General progression of KM-MBFO

- At the point when new group centroid esteems are gotten, new cluster centroid esteems are refreshed in another document and the quantity of cycles is expanded by one.
- This procedure is rehashed until no more changes in group centroid esteems are found, and this state is alluded to as convergence. The last yield clusters are warehoused in an output file record.
- The outcomes have been dissected dependent on different sizes of data sets; better outcomes have been accomplished with expanding data set size.

IV. IMPLEMENTATION AND ANALYSIS & RESULTS

The Hadoop cluster is a special parallel computational cluster that includes a master node and several slave nodes. For a given data set, a file is split into numerous components equal to the block size set for the HDFS cluster. Several experiments were conducted on various numbers of files in the data sets to evaluate the quality and scalability of our proposed algorithms.

To evaluate the performance of the KM-MBFO algorithm, we compared it to other algorithms [21] (e.g., standard K-means, KM-HMR, BFO,PSO). Here we simulated the data set in [22] for performance analysis.

The KDD-99 training data set consists of 4 292 637 instances of 41 dimensional vectors and is labelled data that specifies the attack type (normal or attack). KDD-99 has 22 simulated attack types, which fall in one of the following four categories.

- Denial of service attack
- Users to root attack
- Remote to local attack
- Probing attack

The experiments were carried out on a system with Intel Core i3 CPU M 380 @ 2.53 Ghz and 4GB RAM running Window 8 Professional 64-bit operating system. We have performed experiments to various clustering algorithms and file sizes for the given data set.

We selected this data set due to the reason that clustering makes it easier for grouping of attacks according to their similarities.

All of the proposed algorithms were run with attack data sets of different sizes taken from the above data set to generate effective clustering results. This data set was used to investigate and explore documents to conduct a cluster analysis of unstructured data with better execution times.

4.1. Results and Discussion

The proposed hybrid approach could be a useful primitive for handling Big data sets.

Two phases are involved in the KM-MBFO: HDFS and MapReduce. HDFS stores the Big data sets and is the primary storage system used by Hadoop application.

HDFS is designed to enable high throughput, which enables parallel computation of Input Split blocks. MapReduce phase processes these blocks or splits into map and reduce tasks. The efficient outcome achieved by our proposed approach is depicted below.

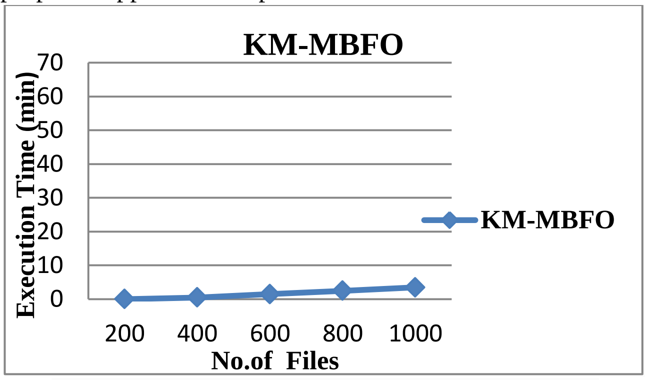

Figure 3: Performance of KM-MBFO based on execution time

Figure 3shows the execution times of the proposed algorithms for document sets of the dataset. As the number of documents to be clustered increases, the time dedicated exceeds for the proposed to KM-MBFO solution. It is noted that the execution time of 600 files are of about 1.5 min. Parallel processing is essential in processing large volumes of data. In this experiment, we only considered some documents of a subset of the data set.

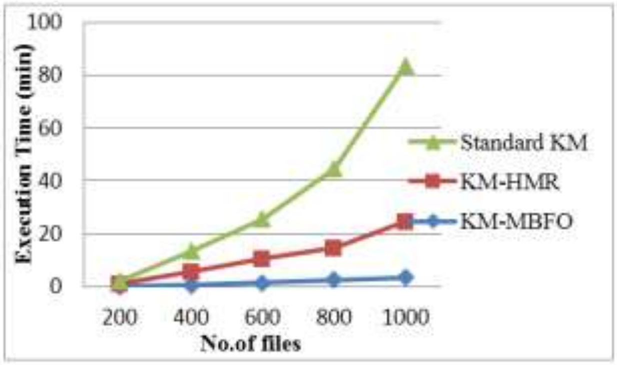

Figure 4: Comparative results based on Execution time

Figure 4 compares the performance of the proposed model to that of the KM-HMR and standard KM algorithms when applied to the attack data set. Parallel K-means worked well with the data sets studied over a reasonable amount of time due to inherent levels of data parallelism. KM-MBFO performed better in terms of execution time, as clusters formed in less time than when using the other existing clustering techniques. Average lengths of time used for each cluster indicate that with parallelization, the proposed clustering algorithm was more efficient than the single node standard K-means clustering method.

V. CONCLUSION

Clustering high dimensional data is one of the strongest problems since it deals with several numbers of attributes in a given data set. This paper proposes the Big data set in Hadoop environment which can be stated as an example of high dimensional data. Enhancement to the K-Means Hadoop Map Reduce algorithm using Euclidean distance based clustering is considered with the goal of producing quality clusters. This paper also deals the K value optimization by means of a Modified Bacterial Foraging Optimization Algorithm that is used in enhancing the Hadoop performance and optimizes the execution time to meet the requirement of handling data sets. Through extensive experimental results and comparative analysis, we evaluate the performance of the proposed KM-MBFO scheme which is superior to the existing approaches.

REFERENCES

1. X. Cui, P. Zhu, X. Yang, K. Li and C. Ji, "Optimized big data K-means clustering using MapReduce", *The Journal of Supercomputing*, vol. 70, no. 3, pp. 1249-1259, 2014.
2. A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Zomaya, S. Foufou and A. Bouras, "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis", *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 3, pp. 267-279, 2014
3. T. Sajana, C. Sheela Rani and K. Narayana, "A Survey on Clustering Techniques for Big Data Mining", *Indian Journal of Science and Technology*, vol. 9, no. 3, 2016.
4. R. Alguliyev, R. Aliguliyev, A. Bagirov and R. Karimov, "Batch clustering algorithm for big data sets", 2016 IEEE 10th *International Conference on Application of Information and Communication Technologies (AICT)*, 2016.
5. V. Eluri, M. Ramesh, A. Al-Jabri and M. Jane, "A comparative study of various clustering techniques on big data sets using Apache Mahout", 2016 3rd *MEC International Conference on Big Data and Smart City (ICBDSC)*, 2016.
6. Q. Zhang, C. Zhu, L. Yang, Z. Chen, L. Zhao and P. Li, "An Incremental CFS Algorithm for Clustering Large Data in Industrial Internet of Things", *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1193-1201, 2017.
7. V. T., "Performance based analysis between k-Means and Fuzzy C-Means clustering algorithms for connection oriented telecommunication data", *Applied Soft Computing*, vol. 19, pp. 134-146, 2014.
8. Y. Li, G. Yang, H. He, L. Jiao and R. Shang, "A study of large-scale data clustering based on fuzzy clustering", *Soft Computing*, vol. 20, no. 8, pp. 3231-3242, 2015.
9. Q. Zhang and Z. Chen, "A weighted kernel possibilisticc-means algorithm based on cloud computing for clustering big data", *International Journal of Communication Systems*, vol. 27, no. 9, pp. 1378-1391, 2014.
10. W. Bi, M. Cai, M. Liu and G. Li, "A Big Data Clustering Algorithm for Mitigating the Risk of Customer Churn", *IEEE Transactions on Industrial Informatics*, vol. 12, no. 3, pp. 1270-1281, 2016.
11. R. Tang and S. Fong, "Clustering big IoT data by metaheuristic optimized mini-batch and parallel partition-based DGC in Hadoop", *Future Generation Computer Systems*, 2018.
12. Z. Benmounah, S. Meshoul, M. Batouche and P. Lio', "Parallel Swarm Intelligence Strategies for Large-scale Clustering based on MapReduce with Application to Epigenetics of Aging", *Applied Soft Computing*, 2018.
13. D. Kumar, J. Bezdek, M. Palaniswami, S. Rajasegarar, C. Leckie and T. Havens, "A Hybrid Approach to

Clustering in Big Data", *IEEE Transactions on Cybernetics,* vol. 46, no. 10, pp. 2372-2385, 2016.

14. C. Sreedhar, N. Kasiviswanath andP. C. Reddy "Clustering large datasets using K-means modified inter and intra clustering (KM-I2C) in Hadoop." *Journal of Big Data,* vol. 4, no. 1, pp.27, 2017.
15. M. S. Hidri, M. A. Zoghlami and R. B. Ayed, R.B. "Speeding up the large-scale consensus fuzzy clustering for handling Big Data", *Fuzzy Sets and Systems,* 2017.
16. D. Feldman, M. Schmidt and C. Sohler, "Turning Big data into tiny data: Constant-size coresets fork-means, PCA and projective clustering", *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms,* pp. 1434-1453, 2013.
17. N. Bharill, A. Tiwari and A. Malviya, "Fuzzy Based Scalable Clustering Algorithms for Handling Big Data Using Apache Spark", *IEEE Transactions on Big Data,* vol. 2, no. 4, pp. 339-352, 2016.
18. K. Peng, V. Leung and Q. Huang, "Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System Over Big Data", *IEEE Access,* vol. 6, pp. 11897-11906, 2018.
19. J. Li and H. Liu, "Challenges of Feature Selection for Big Data Analytics", *IEEE Intelligent Systems,* vol. 32, no. 2, pp. 9-15, 2017.
20. D. Vanisri, "A Novel Fuzzy Clustering Algorithm Based on K-Means Algorithm", *International Review on Computers and Software (IRECOS),* vol. 9, no. 10, p. 1731, 2014.
21. X. Yan, Z. Zhang, J. Guo, S. Li and S. Zhao, "A Modified Bacterial Foraging Optimization Algorithm for Global Optimization", *Intelligent Computing Theories and Application,* pp. 627-635, 2016.
22. M. Wan, L. Li, J. Xiao, C. Wang and Y. Yang, "Data clustering using bacterial foraging optimization", *Journal of Intelligent Information Systems,* vol. 38, no. 2, pp. 321-341, 2011.
23. M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the KDD'99 CUP data set", In Proc. 2nd *IEEE Symp. Comput. Intell. Conf. Security Defense Appl. (CISDA),* vol. 40. Ottawa, ON, Canada, 2009, pp. 44–47.