Firefly Algorithm based Map Reduce for Large-Scale Data Clustering

Siva Krishna Reddy, Pothula Sujatha, Prasad Koti

Abstract*: The technological advancement plays a major role in this era of digital world of growing data. Hence, there is a need to analyse the data so as to make good decisions. In the domain of data analytics, clustering is one of the significant tasks. The main difficulty in Map reduce is the clustering of massive amount of dataset. Within a computing cluster, Map Reduce associated with the algorithm such as parallel and distributed methods serve as a main programming model. In this work, Map Reduce-based Firefly algorithm known as MR-FF is projected for clustering the data. It is implemented using a MapReduce model within the Hadoop framework. It is used to enhance the task of clustering as a major role of reducing the sum of Euclidean distance among every instance of data and its belonging centroid of the cluster. The outcome of the experiment exhibits that the projected algorithm is better while dealing with gigantic data, and also outcome maintains the quality of clustering level.

***Index Terms*: Hadoop, Map Reduce, Clustering, Firefly.**

I. INTRODUCTION

Due to the rapid growth in universal data, the utilization of data mining[1], known as computing procedure is introduced to analyze and discover different patterns out of various perspectives in huge volume of datasets and creating enhanced decisions. It is noted as one of the significant and upcoming necessity in various domains relating to computing. This is due to the difficulty and impossibility of a human being to draw meaningful conclusion within few moments. To automate the process of data analysis, there is aneed for efficient techniques of data mining.

Clustering is a process of assigning the items which reside in a set to target groups otherwise clusters in order to make maximum similarities amongst the members when it is in a similar cluster. It exhibits minimum similarities amongst the members when it is in different clusters. For each item in the information, the major aim of clustering is to predict accurately the corresponding item to the target cluster.

In the past times, there was large number of techniques that had been projected for resolving the clustering issues[2]. Due to the simple nature of the K-means technique, it is well-known by the researchers. Within a problem space, the outcome of k-means produces sensitive results when comparing to the starting position of the centroid in the cluster. In addition, this technique usually converges with the local optima of the adjacent to the initial point of the search, which mightlead to larger number of iterations to reach the optimal solution globally.

Revised Manuscript Received on July 05, 2019.

Siva Krishna Reddy, Department of Computer Science, Pondicherry University, Puducherry, India.

Pothula Sujatha, Department of Computer Science, Pondicherry University, Puducherry, India.

Prasad Koti, Department of Computer Science, Saradha Gangadharan College, Puducherry, India.

In the previous years, for resolving optimization issues in variety of scientific and engineering fields, the researchers have introducedseveral bio inspiredalgorithms. Specially to resolve clustering issues, there exist a large number of methods which depends on the evolutionary computation like genetic algorithm (GA), differential evolution (DE), ant colony optimization (ACO), in addition to particle swarm optimization (PSO). A GA-based method known as "GAclustering" has been projected by Maulik and Bandyopadhyay [3]. Within the feature space, to search suitable centroids of a cluster, the search capability methods of GA can be used. Das et al. [4] projected DE depended methodology for clustering. By employing the ACO technique to resolve clustering issues,Shelokar et al. [5] uses distributed agents that replicate the method, where real ant finds the shortest paths to a food out off the nest. Cura [6] built a model which employs the features of bird flocking otherwise fish schooling activities using PSO based algorithm. This method can be implemented either if the count of cluster that has to be formed is known or unknown. Even though, it is a critical problem when to cluster a huge dataset which comprise of large files with gigantic information. However, by means of computational cost in terms of storage and time, many of the sequential methods of clustering are high-priced. Keeping these issues in mind, there is necessity for a clustering algorithm which employs parallel and distributed computing to access the huge volume of data. Over the previous years, the programming paradigm of MapReduce is popular and an alternate method for parallel programming over data. It is interesting solution for the issue of large-scale clustering because it automatically parallelizes the assignments and thus give a load balancing and fault tolerant model. In distributed environment of computing, a variety of methods depending on MapReduce model had been projected for huge volume of data clustering. In large-scale problems, by transferring the GA primitives of mapreduce and to describe the scalability of those issues, Verma et al. [7] presents GA by employing MapReduce method. To create clustering methodology suitable for huge volume of data, Zhao et al. [8] tailored the k-means method of parallelization in anenvironment of MapReduce. Wang et al. [9] presented a method named parallel K-PSO to develop the outcome of clustering in addition to get better the capacity of gigantic quantity of processing of data by merging the K-means and PSO methods depended over MapReduce. To avoid the inefficacy of PSO algorithm of clustering over huge datasets,Aljarah and Ludwig [10] commenced a technique known as scalable MR-CPSOthrough employing the parallel MapReduce technique. In a distributed environment of computing to deal with massive amount of data, k-NN method depending on MapReduce

model had been projected by Anchalia and Roy [11].

In this work, for huge volume data clustering, a MapReduce-based Firefly known as MR-FF is proposed. With a Hadoop framework, the method Firefly is implemented using the MapReduce model. The main intention is to obtain an optimal solution during the clustering of huge instances of data into cluster groups with a goal of reducing the sum of Euclidean distance among every instance of data and its belonging centroid of the cluster. Consequently, clustering of large-scale information is simplified by the MR-FF method and also the outcome maintains the quality of clustering level.

This paper is ordered as follows. The environmentinformation, together with clustering, FFalgorithm, in addition to Hadoop MapReduce, is representedin brief in Section 2. The proposed model of the distributed MapReduce-based Firefly for data clustering is projected in Section 3. The implementationsetting and outcomes are shown and described in Section 4. At the end, Section 5 present the conclusion.

II. BACKGROUND KNOWLEDGE

A. Clustering

Clustering is a generic word for the kind of unsupervised learning. Clustering is a process of assigning the objects into target groups in order to make maximum similarities amongst the members when it is in a similar cluster otherwise it exhibits minimum similarities amongst the members when it is in different clusters. For distinguishing the similarities among the instances of the data, a distance measure is employed. In each cycle of clustering the data instance are adopted by groups and can be readopted by another cycle till quality level of cluster is obtained, hence clustering is always repetitive process.

In this work, we mainly spotlight the clustering issue that is the count of clusters (*K*)which has to be produced from every data gathering is called as priori [13]. With a view to group the data instances as an evaluationcriteria that are employed in intra-cluster distances. Let us assume $O = \{o_1,o_2,o_3...,o_N\}$ is a collection of *N* instances to be grouped or clustered, $C=\{c_1,c_2,c_3...,c_K\}$ be a collected of *K* centroid of the cluster, moreover let *D* be the count of dimensions in the multidimensional space which describe the objects. For every data instance$o_i$and every centroid of cluster c_j through a vector of *D* range, each for every dimension. The issue of clustering is shown below: For a given *N* instances, assign every instance to each of the clusters (*K)*.The main aim of clustering is to reduce the total number of Euclidean distance which is squared among every instance (o_i) in addition it belong to the cluster's centroid (c_j). The main function is presented as an Eq. (1) below:

$$Min\ D(w,c) = \sum_{i=1}^{N}\sum_{j=1}^{K}\sum_{d=1}^{D} w_{ij}\left|o_{id} - c_{jd}\right| \qquad (1)$$

Wherew_{ij}be the group weight of instanceo_i in cluster *j*, that isw_{ij}be 1 if instance *i* is assigned to cluster *j*, and 0 if it is not. The every centroid of the *j* clusters, $c_j = \{c_{j1},c_{j2},c_{j3},...,c_{jd}\}$, outline the collection of the mean, each and every instancesare acquired to the j^{th} cluster. It may be estimated through Eq. (2) follow:

$$c_{jd} = \frac{1}{N}\sum_{i=1}^{N} w_{ij}o_{id} \qquad ,d = 1,2,3,\dots,D \qquad (2)$$

Where$N_j$be the count of instances in the j^{th} cluster.

The main issue in clustering is to search for an optimal assigning of the element or objects of data to cluster. An optimal solution may be from every probable assignment. The solution enhances when the similarity among intra-cluster is larger.

B. Firefly Algorithm

Initially, FF algorithm is builtthrough Xin-She Yang in 2007 and 2008 on Cambridge University [32, 33], which depended over the activities of fireflies and patterns in flashing. Fundamentally, FA employs the below rules idealized:

- FFs are gender-specific, so that one FFcould be fascinated towards another FFapart from of their gender
- The brightness is relative to the attractiveness, when the distance increases, they together decrease. Hence, for some fireflies which are flashing, the brighter one can be attracted and move to the less bright one. It would arbitrarily move if there is no such a brighter one.
- The brightness of a firefly is unwavering through the landscape of the main function.

We describe the variation in brightnessβ with the distance *r* by

$$\beta = \beta_0 e^{-\gamma r^2}, \qquad (3)$$

As aFF's brightness is relative to the intensity of light seen by neighbouring FF, Whereβ_0 be the attractiveness at *r* = 0.The movement of aFF*i*is fascinated to a furtherattractive (brighter) FF*j* isdescribedthrough

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}\left(x_j^t - x_i^t\right) + \alpha_t \epsilon_i^t, \quad (4)$$

Where the second term is regarding to the attractiveness, $\epsilon_i^t$is a vector of arbitrary counts drawn as of a Gaussian distribution otherwise uniform allocation at time period*t*. It is simple arbitrary walk if $\beta_0 = 0$. In other way, it diminishes to the PSO variant if the value $\gamma = 0$ [32]. In addition, the extending of randomization ϵ_i^t is simple to distribution methods like Levy flights [32].

C. Hadoop MapReduce

Hadoop is a dispersed massive data analysis environment and built to deal with gigantic task over a collection of machines[15]. It is very suitable for various processes like task scheduling, concurrency control, resource management, and also for node failure recovery. Hadoop Distributed File System (HDFS), which is a hadoop's file system which employs MapReduce programming structure which is very flexible and capable to analyse the data even the size is in petabytes.

MapReduce is a theory for massive processing of information in a parallel programming way. The structural design of the MapReduce parallel programming model is depicted in figure. 1. Over numerous machines, the MapReduce program analyses loads of data. MapReduce divides the data into independent chunks in

additionto the volume of every division, which is a function of the sum of information and the count of computing nodes that are accessible. The data that are processed through a MapReduce is in the structure of key/value pairs.

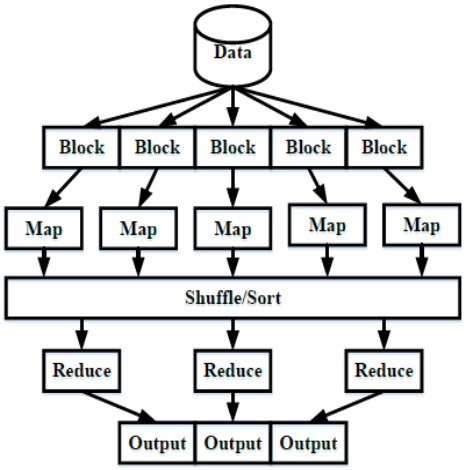

Fig. 1. MapReduce parallel programming model

The fundamental scheme of MapReduce is split into two major steps: Map and Reduce phase. In the initial step, for every input of key/value pair, the map function is invoked and it produces output key/value pairs as the intermittent outcomes. While in the second step, every intermittent outcomeis combined or grouped through keys. For every key and their associated values, the reduce process is called once and creates the outcome ranges as an end result. The major differences among these two phases arethat the map function is intended for decomposition of the task and Reduce is modelled for combining the outcomes.

III. ADOPTING THE MAPREDUCE-BASED FF FOR DATA CLUSTERING

There are two major process for modifying the cluster centroid and evaluation of the fitness over a massive data in the method of MR-FF. The modification in the cluster's centroid is based on the FF as denoted in section 2.2. The main aim is to search for a universal optimal solution of Euclidean distance which is squared among every instance and the cluster's centroid. On the other hand, we take steps to search optimal assigning of the elements otherwise data objects to groups which reduces the total Euclidean distance which are squared in Equation1, that is employed as a fitness function in FF. Acquiring of MapReduce-based FF for data clustering, it is demonstrated in figure 2.

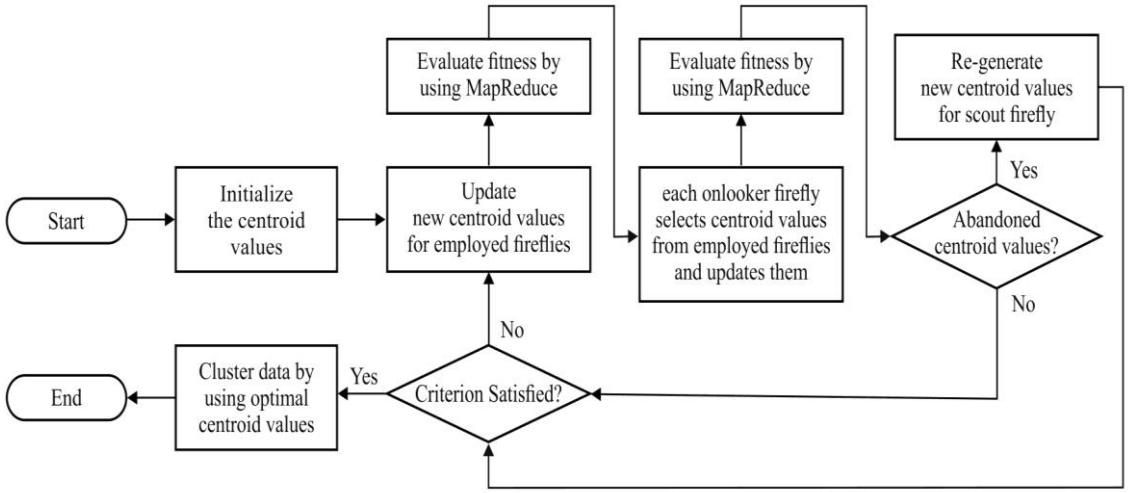

Fig. 2.Using the Map Reduce-based FF in finding the optimal centroid values

As shown above in figure 2, the centroid values are produced. These parameters are assumed as the solution for the engaged FF. One of the metrics that is used to estimate the quality of the cluster and fitness of the solution is sum of the squared Euclidean distance. The brighterFF updates as well as share the novel solution that it had found so far with the less brighter FF. The brighter FF chooses the solutions which create an enhanced fitness, and update those into the solution and estimate the fitness of its own solutions again. This procedure is recursive unless and until the count of iterations is as same as the threshold range (MCN). The solution is abandoned if one solution cannot enhance the fitness range within some sort of period and fresh solutions are recreated through the FF. Even though, while dealing with a massive dataset, the time it takes to execute the fitness value is higher. Hence, MapReduce procedure is employed to estimate the fitness value. The operations of MapReduce functions are depicted in Figure 3.

As shown below in figure 3, fetching the cluster centroid out off the FF procedure demonstrated over and the information is stored within the HDFS in the map function. The function filters the centroid values out off every FF and estimates the distance range among information record and the centroid rates, and then proceed to return the minimal distance along its centroid_ID.While for building a fresh composite key as well as a fresh value out off the minimal distance the map function employs the FF_ID with a centroid_ID which have a minimal distance. Subsequently, this function provides a fresh key and fresh rate to the reduce function. It then groups the rates along with similar key to estimate the average distances.

```
Function Map (key: RecordID, value: Record)
Initialization:
record_ID          = key
record             = value
read(Fireflies)
for each firefly in Fireflies
        firefly_ID          = Extract_FireflyID(firefly)
        CV                  = Extract_Centroids(firefly)
        min_Dist            =ReturnMinDistance(record, CV)
        centroid_ID         = i          //ith centroid contains minDist
        new_Key             =( firefly_ID, centroid_ID)
        new_Value           =(mm_Dist)
        Emit(new_Key, new_Value)
end for
end function

function Reduce (key: (firefly_ID, centroid_ID), value_list: (minDist,1))
Initialization:
Count              = 0
sum_Dist           = 0
avg_Dist           = 0
for each value in value_list
        min_Dist           = ExtractMinDist(value)
        count              = count + 1
        sum_Dist           = sum_Dist + min_Dist
end for
avg_Dist           = sum_Dist/count
Emit(key, avg_Dist)
end function
```

Fig. 3. Operations of MapReduce functions

At the end, to build it as a fitness range for every FF, the reduce function provides the key with common distance.

IV. EXPERIMENTAL SETTINGS AND RESULTS

For data clustering, the experimentation and performance of the projected MR-FF technique is provided. This work aimed to achieve better quality level of clustering and effective parallelism in algorithm by estimating the solution. Every function and methods, experimentation was done in Perl and Hadoop cluster of 10 nodes with every one comprising of configuration of a CPU with 2.26 GHz, 2 GB memory, as well as a 120 GB hard disk drive. Ubuntu version 14.04 is the operating system over each and every node along with Apache Hadoop version 2.6.2.

Table I. Dataset description

Dataset	Baseline Dataset	#Records	#Dimensions	#Clusters
I	Iris	10,000,050	3	4
II	CMC	10,000,197	3	9
III	Wine	10,000,040	3	13
IV	Vowel	10,000,822	6	3

With an intention to estimate the performance of our MR-FF technique over massive dataset, the experiment is split into two major phases. In the initial phase, we employ the synthetic datasets as demonstrated in Table 1. Depend upon the baseline datasets fetched out off the UCI Machine Learning Repository [17], we synthesized the information which comprise variety of features by means of count of dimensions as well as clusters. For every baseline dataset to synthesize the information, we just create duplicate the records several times to create huge dataset volumes about 10^7 records. In the second phase, in a disk drive manufacturing procedure, we apply the projected technique to the created dataset.

By employing F-Measure, the quality of the cluster is stated which is a metric out of the information retrieval which employs the ideas of precision and recall[19] to estimate the clustering accuracy. For every query F-measure, each cluster is treated as a query and each class as the expected outcome. The F-measure of cluster *j* produced through the algorithm and the class *i* employed is expressed in Equation (5):

$$F(i,j) = \frac{2 \cdot r(i,j) \cdot p(i,j)}{r(i,j)+p(i,j)} \quad (5)$$

where *r* and *p*indicates recall and precision.

Recall is described as $r(i,j) = \frac{n_{ij}}{n_i}$in addition precision is denoted as$r(i,j) = \frac{n_{ij}}{n_i}$ where n_{ij}be the counts of class *i* items in cluster *j*, whenn_i and $n_j$be the volume of class *i* and cluster *j*, correspondingly. The general F-Measure for the whole dataset of size isacquiredin Equation (6):

$$F = \sum_i \frac{n_i}{n} \max_j (F(i,j)) \quad (6)$$

Where F comprise of upper bound rate of 1.0, that take place only objects of class *i* are acquired by the cluster j and also for also for entire objects of class i. Aparently, a high quality of cluster solution contains a larger rate of the F-Measure.

In the initial section for comparing the quality of the cluster, we compared the outcome of the MR-FF method with PK-Means algorithm [8] in addition to the parallel K-PSO algorithm [9]. Table 2 demonstrates the each dataset's clustering quality outcomes acquired by the algorithms. An average of 0.1% of data record is arbitrarily chosen out off the dataset for estimating initial cluster centroid.

Table II. F-Measure outcomesgained by the algorithms on the synthetic datasets

Dataset	MR-FF	PK-Means	Parallel K-PSO
I	0.942	0.667	0.785
II	0.487	0.298	0.324
III	0.818	0.482	0.517
IV	0.743	0.586	0.627

Over all of the dataset, our algorithm gains sensibly good outcomes demonstrated in Table 2. The proposed algorithm MR-FF shows enhanced F-measure values than PK-Means and the K-PSO parallel techniques thus the projected method produces enhanced quantitative evaluation outcomes have been attained.

In terms of parallel computing, speedup describes that how far the parallel algorithm is speeder than respective sequential algorithm and scalability means to the ability of a machine to enhance the whole throughput underneath the enhanced load. The estimation of the speedup (S_p) estimated through Eq. (7):

$$S_p = \frac{T}{T_N} \quad (7)$$

Where the T is the executing period of one node in addition $T_N$is the executing period of N nodes.

We studied by modifying the count of nodes in every execution multiplying by 2, the scalability of MR-FF method is estimated. The executing times as well as speedup measures are depicted in Fig.6to Fig. 9.We considered the performance scalability of the MR-FFtechnique through altering the count of nodes in each executes through the multiples of 2. F-Measure outcomesover the Synthetic dataset gained by the algorithms for dataset I, dataset II, dataset III, dataset IV is shown in Fig.4 and Fig. 5.

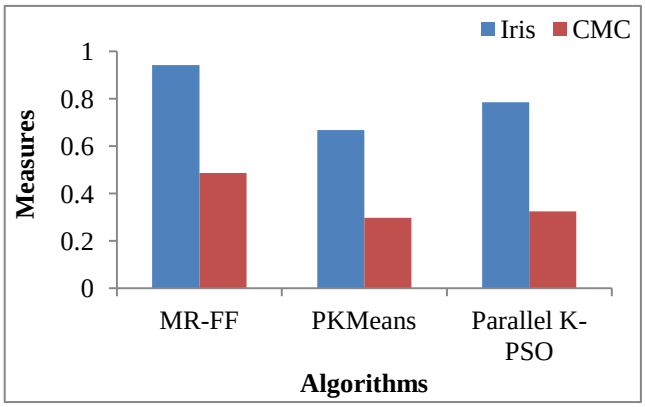

Fig. 4. F-Measure outcome on the Synthetic dataset acquired by the algorithms for dataset I and dataset II

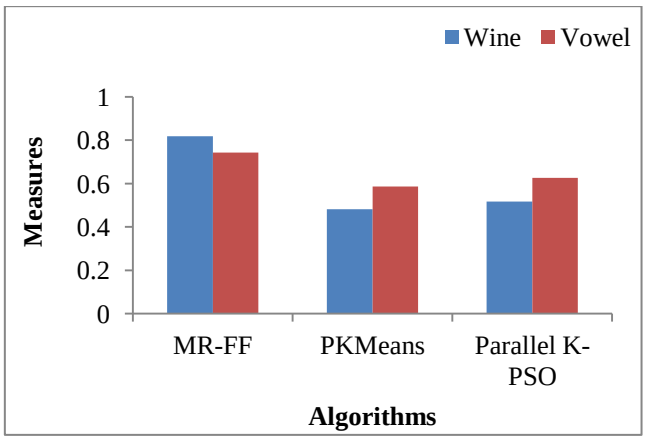

Fig. 5. F-Measure outcomes on the Synthetic dataset acquired by the algorithms for dataset III and dataset IV

Figure 4 and Figure 5 depicts that in the Hadoop cluster, the execution time of MR-FF mitigates linearly with an enhancement in count of nodes. For the dataset that comprise of least count of dimensions, cluster performance is decreased. Even though for a dataset that comprise of greater count of dimension and clusters, the method performance is near to the ideal state. Table 3 demonstrates that to focus the performance scalability over the dataset volumes, the efficacy outcomes in huge dataset volumes with 10 nodes acquired through MR-FF. The efficacy could be estimated through Eq. (8). Make a note that, to create a massive dataset, we duplicate the records multiple time out off 2 GB to 10 GB for every dataset.

$$Efficiency = \frac{S_p}{N} \quad (8)$$

Table III.Efficiency outcomes of variation in dataset volumes with 10 nodesacquiredthrough the MR-FF algorithm over dataset I

	Dataset sizes (GB)				
	2	**4**	**6**	**8**	**10**
F-Measure	0.942	0.942	0.942	0.942	0.942
Efficiency	0.955	0.964	0.968	0.972	0.975

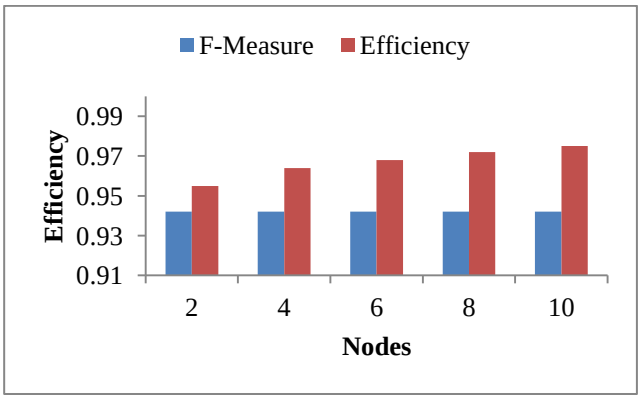

Fig. 6.Execution time and speedup outcomes of MR-FF with 10 Hadoop cluster nodes by dataset I

Table IV.Efficiency outcomes of variation in dataset volumes with 10 nodesacquiredthrough the MR-FF algorithm over dataset II

	Dataset sizes (GB)				
	2	**4**	**6**	**8**	**10**
F-Measure	0.487	0.487	0.487	0.487	0.487
Efficiency	0.974	0.977	0.980	0.985	0.991

Table V.Efficiency outcomes of variation in dataset volumes with 10 nodes acquiredthrough the MR-FF algorithm by dataset III

	Dataset sizes (GB)				
	2	**4**	**6**	**8**	**10**
F-Measure	0.818	0.818	0.818	0.818	0.818
Efficiency	0.982	0.984	0.984	0.992	0.998

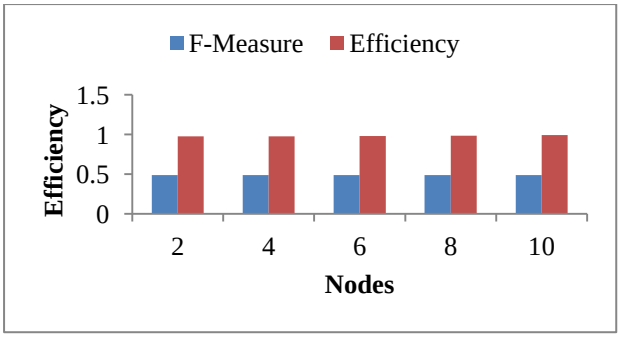

Fig. 7.Executing time and speedup outcomes of MR-FF with 10 Hadoop cluster nodes by dataset II

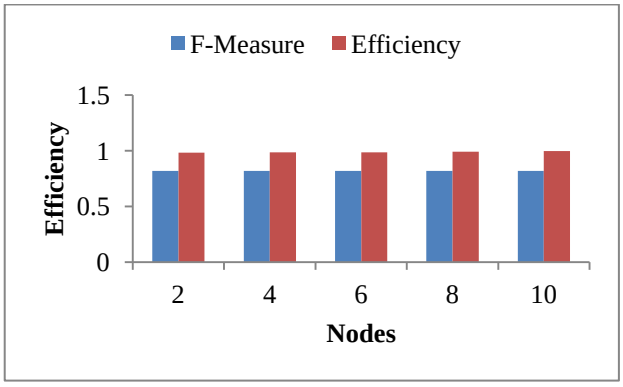

Fig. 8.Executing time and speedup outcomes of MR-FF with 10 Hadoop cluster nodes over the dataset III

Table VI.Executingoutcomes of variation in dataset sizes with 10 nodesacquired by the MR-FF algorithm by dataset IV

	Dataset sizes (GB)				
	2	**4**	**6**	**8**	**10**
F-Measure	0.743	0.743	0.743	0.743	0.743
Efficiency	0.961	0.968	0.974	0.977	0.978

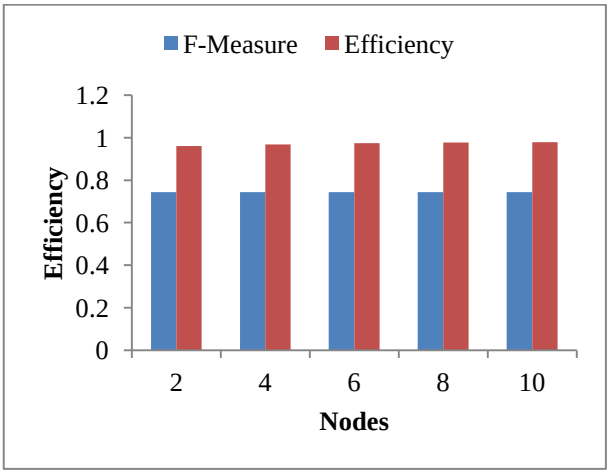

Fig. 9.Running time and speedup results of MR-FF with 10 Hadoop cluster nodes on the dataset IV

Table 3 to Table 6 demonstrates the performance of the MR-FF method enhances as the volume of the dataset increases gigantically. The efficacy was 90% otherwise better. Hence, the algorithm provides scalability as the problem volume enhances. Our outcomes recommend that the computation amount needed to search a solution is affected by the volume of the dataset. The distribution of computation to large nodes is enhanced when the volume of the problem increases.In the second section, using a real huge dataset comprising of features like materials, tools and processes acquired from the hard disk drive manufacturing procedure. The MR-FF technique is estimated and the performance is measured. There are two groups of data: product objects passed (1,457,000 records), and product objects failed (143,000 records). By including attributes and machine parameters, these groups are categorized by 30 characteristics.

Executing time and speedup outcomes of MR-FF with 10 Hadoop cluster nodes over the dataset depicted in Figure 6. For this issue, again MR-FF method gives significant solutions comparing to the other algorithms. However, the enhancement of the common F-measure for the MR-FF technique when comparing with other techniques produces 58%.

Over HDD dataset, the algorithm scalability of our projected technique is depicted in fig.7. By enhancing the count of cluster nodes employed in our algorithm, it leads to drop out in execution time, with no drop out in accuracy.

When summarizing the experimental outcomes, in the problem space, the K-Means algorithm is always especially speed. However, it is too sensible to the locations of the initial centroid of the cluster. The solution acquired through the K-Means function usually combines to the local optima of adjacent to the initial point of the search. While contrasting between the compared techniques, the FF technique is capable to the produce superior results than PSO. The FF method uses together the exploitation as well as exploration in the search procedure, when PSO implement only exploitation depended modification of the velocity of particles. Within FF method, the exploitation is managed through the brighter FFs, when the exploration is managed. The FF will try to arbitrarily find a novel solution again, when some solutions are entangled at any local optima. Moreover, this implementation of MapReduce with FF algorithm mitigates the time and costs in a massive clustering procedure in data. The outcomes recommend that it maintains good cluster quality using distributed algorithm and it is even more suitable to analyse a large-scale data.

V. CONCLUSION

Map Reduce based FF algorithm known as MR-FF for huge dataset clusteringis projected in this work. Within the Hadoop environment, the FF algorithm is experimented on the MapReduce structure and used to improve the procedure of clustering in large-scale data. This work also contains the detailed explanation with the comparison of existing methods such as PK-Means in addition to the parallel K-PSO techniques. To prove the performance of our algorithm depending over the group of the quality of solution and effectiveness of parallel algorithm, the implementation is tested together with synthetic and practical data. The outcomes so acquired out off the projected technique depict that the MR-FF technique provides an enhanced level of quality in clustering as well as performance and it also overcomes the other recent techniques. Thus we can come to a conclusion that the projected technique is bettertogether in quality and in performance. Therefore, it serves a helpful alternating solution for the area in which there is a need to cluster gigantic data.

REFERENCES

1. X. Wu, G.-Q. Zhu, W. Wu, Ding, "Data mining with big data", IEEE Transactions on Knowledge and Data Engineering, vol. 26, 2014, pp. 97-107.
2. R. Xu, D. Wunsch II, "Survey of clustering algorithms", IEEE Transactions on Neural Networks, vol.16 2005, pp. 645-678.
3. U. Maulik, S. Bandyopadhyay, Genetic algorithm-based clustering technique, Pattern Recognition, vol. 33, 2000, pp. 1455-1465.

4. S. Das, A. Abraham, A. Konar, "Automatic clustering using an improved differential evolution algorithm", IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 38, 2008, pp. 218-237.
5. P.S. Shelokar, V.K. Jayaraman, B.D. Kulkarni, An ant colony approach for clustering, Analytica Chimica Acta, vol. 509, 2004, pp. 187-195.
6. T. Cura, A particle swarm optimization approach to clustering, Expert Systems with Applications, vol. 39 2012, pp. 1582-1588.
7. A. Verma, X. Llora, D.E. Goldberg, R.H. Campbell, Scaling genetic algorithms using MapReduce, in: Proceedings of 9th International Conference on Intelligent Systems Design and Applications, 2009, pp. 13-18.
8. W. Zhao, H. Ma, Q. He, Parallel k-means clustering based on MapReduce, in: CloudCom, 2009, LNCS, vol. 5931, 2009, pp. 674-679.
9. J. Wang, D. Yuan, M. Jiang, Parallel K-PSO based on map reduce, in: Proceedings of IEEE 14th International Conference on Communication Technology (ICCT), 2012, pp. 1203-1208.
10. I. Aljarah, S.A. Ludwig, Parallel particle swarm optimization clustering algorithm based on MapReduce methodology, in: Proceedings of 4th World Congress on Nature and Biologically Inspired Computing (NaBIC), 2012, pp. 104-111.
11. P.P. Anchalia, K. Roy, The k-nearest neighbor algorithm using MapReduce paradigm, in: Proceedings of 5th International Conference on Intelligent Systems, Modelling and Simulation, 2014, pp. 513-518.
12. A. Jain, M. Murty, P. Flynn, Data clustering: a review, ACM Computing Surveys, vol. 31, 1999, 264-323.
13. A. Banharnsakun, B. Sirinaovakul, T. Achalakul, The best-so-far ABC with multiple patrilines for clustering problems, Neurocomputing vol.116, 2013, pp. 355-366.
14. D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensive survey: artificial bee colony (ABC) algorithm and applications, Artificial Intelligence Review, vol. 42, 2014, pp. 21-57.
15. K. Shvachko, H. Kuang, S. Radia, R. Chansler, The hadoop distributed file system, in: Proceedings of the IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), 2010, pp. 1-10.
16. J. Dean, S. Ghemawat, MapReduce: simplified data processing on large clusters, Communications of the ACM, vol. 51, 2008, pp. 107-113.
17. C.L. Blake, C.J. Merz, UCI repository of machine learning databases [Online], Available: http://archive.ics.uci.edu/ml/ datasets.html [1998].
18. U. Taetragool, T. Achalakul, Method for failure pattern analysis in disk drive manufacturing, International Journal of Computer Integrated Manufacturing, vol. 24, 2011, pp. 834-846.
19. G. Hripcsak, A.S. Rothschild, Agreement, the F-measure, and reliability in information retrieval, Journal of the American Medical Informatics Association vol. 12, 2005, pp. 296-298.
20. A. Grama, A. Gupta, G. Karypis, V. Kumar, Introduction to parallel Computing, Addison-Wesley, USA, 2003.

AUTHORS PROFILE

Siva Krishna Reddy, Research Scholar in Pondicherry University, Finished Diploma in Computer Science and Engineering, B.Tech in Computer Science and Engineering, M.E in Computer Science and Engineering. Research areas includes Big Data and Optimization Techniques

Pothula Sujatha is an Assistant Professor in the Department of CSE, Pondicherry University (PU), India. Previously, she received her MTech and PhD in Computer Science and Engineering from PU in 2005 and 2012, respectively. Her research interests include wireless sensor networks, information systems, and performance evaluation. She holds three books in information retrieval and cloud computing, and has published several research articles including more than thirty-five proceedings and journal publications. She serves for the Technical Committee of International Arab Journal of e-Technology

Prasad Koti working in Saradha Gangadharan College, Pondicherry(PU). Currently, he is working in the fields of optimization algorithms and evolutionary computing.