Webology (ISSN: 1735-188X)
Volume 21, Number 1, 2024

Evaluation Of The Efficiency Of Clustering Using Ik-Means And Imap-Reduce Approach For Microarray Data

Araja Raja Gopal[1] , Dr.M.H.M.Krishna Prasad[2]

[1]Research Scholar, Department of Computer Science, Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, India.

[2]Professor, Department of Computer Science and Engineering, University College of Engineering Kakinada,Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, India.

Abstract— Clustering algorithms are part of algorithms of unsupervised-learning that are commonly used in different fields. Two major impacts on the data clustering algorithm have been the rapid developments and creation of electronic data. This includes the view of how the big data is stored as well as how it is processed. A cluster has a high level of resemblance to the same cluster and a low level of resemblance to other clusters. Clustering algorithms are commonly used in all fields, such as retail, banking, development, etc.. Even though different methodologies are proposed with different scenarios, but none of the existing methodologies are proven to be a better approaches. Here, in our work we have adopted an improved Map-Reduce programming model to combine the Canopy clustering and K-means clustering algorithms to process the Microarray data with the available commodity hardware with an aim to achieve better performance. The results obtained from different scenarios shown that the proposed method was capable of improving computational speed significantly by increasing the nodes as required. In this research, this paper explored the efficiency by evaluating and implementing the proposed Improved k-means with Improved Map Reduce algorithm which runs on Hadoop Frame work using Microarray dataset along with different datasets.

Keywords— Map-Reduce, K-Means, Improved K-Means, Canopy Clustering, Hadoop, Parallel Computing, Distributed Computing.

I. INTRODUCTION

From centralized architecture to distributed architecture, the model of processing huge datasets has been changed. They also found that data cannot be processed using any of the new centralized architectural methods as companies face problems with accessing large chunks of data. The companies faced problems such as productivity, better performance and large amount of resource costs with the processing of data in the centralized way of environment in addition to time constraints with the support of distributed delivery [1]. The Hadoop Project from Apache is considered as one of the available best open source frameworks available in the present market which uses the distributed architecture for solving problems for processing the data. The popular Hadoop Architecture of Apache is used to implement various elements, like data clusters, map-reduction algorithms as well as distributed processing. Based on location, various complex data problems can be solved and provide the necessary information obtained back into the system hence increasing the user experience as the system performance [2].

UNSUPERVISED LEARNING AND CLUSTERING PROCESS

Clustering algorithms are a component of algorithms for unsupervised machine learning. Why are they unsupervised? The goal variable is not present because of this. The model is trained on the basis of some

input variables that aim to find intrinsic classes (or clusters). We can't name those classes because the goal variable is not present. Algorithms for clustering are commonly used in all fields, such as retail, banking, manufacturing, healthcare, etc. Companies use them in corporate terms to distinguish clients[3]. In healthcare, for example, a hospital could cluster patients depending on their tumor size in order to handle patients of different tumor sizes differently. This approach helps us organize unstructured knowledge. It can be used for tabular data, pictures, text information, etc.

Types of Clusering:

Clustering algorithms are of several varieties, for example as K means, fuzzy c- means, hierarchical clustering, etc. Other than these, several other techniques have been developed that are only used for particular data sets or forms (categorical, binary, numeric).There are numerous clustering behaviours between these different clustering algorithms, as seen in the figures below. 1(a) & 1(b) respectively.

A. Soft Clusering

Soft clustering (also understood as soft k-means or fuzzy-clustering) is a kind of clustering in which more than one data points may belong to more than one cluster (i.e., clusters may overlap)

B. Hard Clustering

Hard Clustering: In exactly one cluster, an observation/data point is separated (i.e., clusters do not overlap).

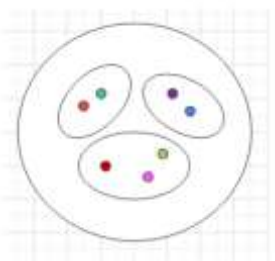

Fig.1(a) Hard Cluster

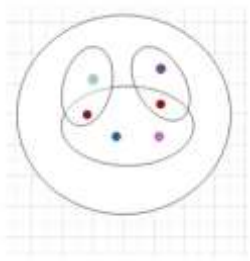

Fig.1(b) Soft Cluster

C. Canopy Clustering

It can be viewed as an unsupervised pre-clustering algorithm that can be used prior to clustering with K-means or Hierarchical clustering. In the case of large data sets, where a direct implementation of the main algorithm may be impractical due to the scale of the data set, it is essentially done to speed up the clustering. Canopy clustering is a form of rapid and approximate clustering. The input data points are divided into overlapping clusters called canopies [4].

D. K-means for Clustering

At the beginning of the algorithm of K-Means, random initialization (different initializations) of centroids will lead to dissimilar clusters as the algorithm of k-means may conform to a local optimum but may not converge to a global optimum.

The canopy clustering method has been adapted to address this issue [5]. It assigns all the data points to the cluster in the manner that the sum obtained of square distance between those data points and the cluster's center (the arithmetic mean of all the data points considered belonging to the cluster) is at least the square distance between the data points and the cluster's center [6]. If there is less variation between the clusters, there is more homogeneity among the data points are found in the same cluster (similar).Most of the clustering algorithms function differently, depending on the features of the considered data set and all the initial assumptions for defining classes [7]. Therefore, in most applications, the resulting clustering scheme requires some form of evaluation with regard to its performance, accuracy and validity. The purpose of this paper was to test and evaluate the proposed algorithm as well as compare with the existing clustering algorithms.

II. BACKGROUND STUDY

Parallel computing vs Distributed computing

The key difference between these two is that a parallel computing system consists of several processors that use a shared memory to communicate with each other, whereas a distributed computing system involves multiple processors linked by a communication network, as shown in the figure 2(a) and 2(b) below [8].

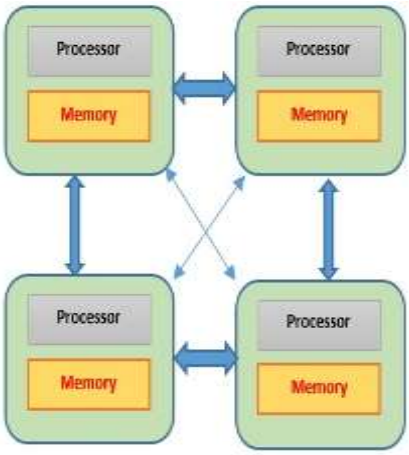

Fig.2(a) Distributed Computing

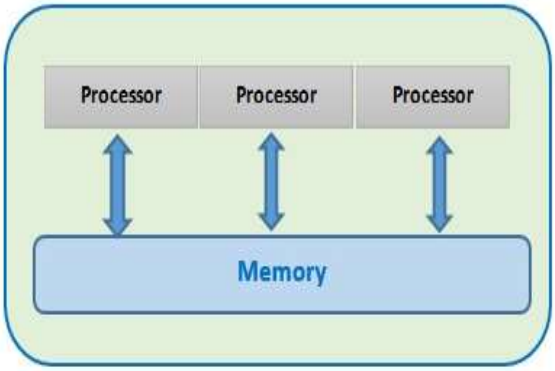

Fig.2(b) Parallel Computing

Issues Influencing the Distributed and Parallel Computing:

- Dependency Between Processes
- Which is More Scalable?
- Resource Sharing
- Synchronization
- Where Are They Used?

A. Introduction to Hadoop Framework:

There are several components in the Hadoop architecture, the most notable of which are Hadoop Distributed File System in short HDFS and the programming model for Map-Reduce. Hadoop has high performance, high reliability, good fault tolerance and low cost of operation and maintenance deployment characteristics[9].

The Hadoop system utilises the programming paradigm of Map Reduce to process large data by distributing and aggregating data through a cluster. One of the approaches used to process big data hosted in large clusters is Map-Reduce. In this approach, jobs are processed by breaking into small parts and spreading across nodes. The execution time of jobs is influenced by parameters such as the distribution method over nodes, the number of jobs kept in parallel, as well as the number of nodes involved in cluster. The aim of this paper is to decide how the available number of nodes, maps and reductions affect the output/performance of the Hadoop structure[10].

B. K-means Clustering:

Since distance-based measurements are used by k-means to evaluate the similarity between data points, data standardization is needed to ensure mean=0 and standard deviation=1. And the attributes of each dataset will have different measurement units, such as age vs. income.
For convergence, more iterations(more communication) happen. As K increases, speed-up decreases. The method of k-means is using the approach to solve the issue understood as the **Expectation-Maximization** [11]**.**
The objective function is:

$$J = \sum_{p=1}^{m} \sum_{q=1}^{n} W_{pq} \left| x^p - \mu_q \right|^2$$

Where, w_{pq}=1 for the data point x_i, and it belongs to the cluster k and if $w_{pq} = 0$. Also, μ_q was considered as the centroid of cluster x^i.
The E-step is to assign the available data points to the closest cluster and the M-step is for computing each cluster's centroid.

C. E-step:

If we reduce the value of J, w.r.t. w_{pq} and treat the μ_q is first fixed. Then the value of J w.r.t. μ_q is reduced and w_{pq} is fixed and treated. And technically speaking, first distinguish J w.r.t. w_{pq} and do the updation of those assignments to the cluster.

$$\frac{\partial j}{dw_{pq}} = \sum_{q=1}^{m} \sum_{q=1}^{K} \left| x^p - \mu_q \right|^2 \quad => w_{pq} = \left\{ \frac{1}{0} \text{ if } k = \text{argmin}_j \left| x^p - \mu_q \right|^2 \right\}$$

The assignment of the data point x_i to its nearest cluster basing on the value of sum of its square distance from the cluster's centroid .

D. M-step:

After the cluster assignments are over, then differentiate the J w.r.t. μ_q and recompute those centroids obtained in the previous stage.

$$\frac{\partial J}{\partial \mu_q} = 2 \sum_{p=1}^{m} w_{pq} (x^p - \mu_q) \quad => \quad \mu_q = \frac{\sum_{p=1}^{m} w_{pq} x^p}{\sum_{p=1}^{m} w_{pq}}$$

It means that the centroid of every cluster is recomputed to represent new tasks.

III. RELATED WORK

Different authors [12] [13] [14] [15] [16] [17] proposed different methodologies to explore the working and performance of them.

The summary of different clustering techniques have been shown in the table. 1(a) & 1(b) below [18].

Summarization of different clustering techniques:

Category	Algorithm	Dataset Type
Partitioning Based Clustering	K-Means	Numerical
	K-Modes	Categorical
	K-Medoids	Categorical
	CLARA	Numerical
	PAM	Numerical
	CLARANS	Numerical
Hierarchical based clustering	BIRCH	Numerical
	CHAMELEON	All types of data
	PDDP	Numerical
	DHCC	Categorical
Density based Clustering	DBSCAN	Numerical
	DENCLUE	Numerical
	OPTICS	Numerical
	DBCLASD	Numerical
Grid-based clustering	STING	Special Data
	CLIQUE	Numerical
	OPTIGrid	Special Data
Model based clustering	EM	Special Data
	COBWEB	Numerical
	MCLUST	Numerical
Feature selection	CFS	Alphanumerical
	SFS	Numerical
	MBF	Numerical
Feature Extraction	PCA	Numerical
	LDA	Categorical
	SVD	Numerical

Table 1(a). Summary of different clustering techniques

Category	Advantages	Disadvantages
Partitioning Based Clustering	• Easy to implement • Can easily process large datasets.	• Noise-responsive and data outliers • Not ideal for clusters which are non-spherical • A randomly constructed cluster centre problem • Cluster efficiency depends on the amount of these user-introduced parameters.
Hierarchical based clustering	• Less immune to noise and outliers • Appropriate for any form of feature.	• Sensitivity to the criterion for scalability • It can't be reversed if a process is performed.
Density based Clustering	• Resilient against outliers • Automatically measure the number of clusters	• Cluster output depends on the threshold range.

	• Shapes uninformed type clusters.	• For high-dimensional datasets, unacceptable
Grid-based clustering	• Speedy Handling Time Self-administration of the number of data items	• Cluster quality depends directly on the number of cells in each dimension added by the consumer.
Model based clustering	• More adaptable to data on noise and outliers • Quick speed of manipulation • The number of clusters is determined automatically.	• Multifaceted in nature.
Feature selection	• Reducing the Dataset • Redundant, irrelevant or noisy data is eliminated.	• Do not have an effective high-dimensional dataset solution compared to feature extraction • It should be carried out before the classification algorithm is applied.
Feature Extraction	• Reduce the dataset, maximise the cost of care, • Rapid speed of handling and more acceptable for scalability.	• It should be carried out before the classification algorithm is applied. • Loss of interpretability of data.

Table 1(b). Summary of different clustering techniques

IV. PROPOSED METHODOLOGY

A. Overview

The proposed algorithm operates in the two stages: the first stage uses Canopy Clustering is to produce initial centroids, and the second stage uses the first stage centroids to get the final set of clusters shown in Figure.3 below. Unlike the simple K-Means, where the initial seed selection process is random, the strategy of Canopy Clustering is used in our approach to pick better seeds to minimize the number of rounds taken to converge.

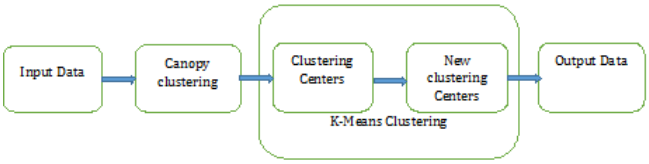

Fig.3. Proposed Methodology Overview

The variation in the sizes of the dataset helps to assess the efficiency of the proposed algorithm. With the same datasets, the k-means sequential algorithm is checked and device configuration and performance variations are presented and evaluated between that sequential and proposed k-means.

The operation of the proposed improved K-means algorithm is mentioned here. The fundamental motive of our work is to gracefully deal with enormous quantities of data without degrading the efficiency of clustering. The basis of our work is the K-means algorithm which gives the output of a group of clusters. Initial knowledge is split into small partitions and distributed in a computer cluster between different nodes. Data chunks are stored in the HDFS. The HDFS holds 3 replicas of data chunk (default), which minimizes the chances of loss of data due to node failure. According to the programmer's request, the

replication factor may be increased or reduced. In our proposed work to handle large quantity of the data, simple algorithm of K-Means is optimized. Here, description of our suggested approach is shown in Fig. 4 from below.

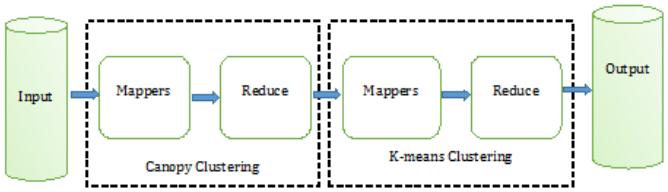

Fig.4. Illustration of Proposed - Canopy and the K-means algorithm with map-reduce

B. Initial Centroids Selection Strategy (Canopy Clustering):

In the output of the parallel K-means algorithm, the initial centroids play an important role. Various initial centroids can also contribute to various outcomes of clustering and different efficiencies. The researchers have therefore proposed all kinds of enhanced methods to avoid this sensitivity. By a specialized approach that is far removed from each other than the random centroids, it is far easier to pick k centroids.
A Canopy clustering approach is then used to classify the initial centroids as pre-clustering. The first point is chosen randomly in canopy clustering and the remaining points are chosen based on the following proposed canopy clustering Algorithm-1.

Algorithm-1: **For Improved Canopy Clustering (Proposed): (Map-Reduce Approach)**

Map Algorithm:
(1) As a main and vector point as a value, the centroid points obtained from the generation of Canopy are considered.
(2) Use the Distance Calculation Function shown in the Algorithm-2 to compute the distance between the centroid point & the data point by using (1)
(3) Using each data point's membership value compute (2).
(4) Build a matrix for membership.
(5) Clusters are constructed using the nearest centroid and the data points allocated to that particular cluster.
(6) Preserves the information on which data point is located in which cluster.

Reduce Algorithm:
1) For each cluster, recalculate the centroid.
The given threshold value for the reduction process activity is significantly greater than the value of the method-map. So, to minimize the number of reduction processes, the node requires only a reduction node pair from all maps to get the canopy center of the merger set.

1) Data is managed into an acceptable input format
2) Each mapper is performing canopy clustering on its input set of points and outputs the centres of its canopies
3) In order to create the final canopy centres, the reducer clusters the canopy centres

Each mapper processes a subset of the total points during the map stage and applies the chosen measure of distance and thresholds to generate canopies. In the mapper, an internal list of canopies will be attached to each point found to be within an existing canopy. After observing all of its input vectors,

the mapper updates all of its canopies and normalizes their sums using a constant key ('centroid') to a single reducer to generate generated canopy centroids.
All initial centroids are received by the reducer and the canopy calculation and thresholds are again applied to establish a final collection of canopy centroids (i.e. clustering the cluster centroids). The output format of the reducer is: Sequence File (Text, Canopy) with the canopy recognition encoding value.

So, the desired clustering is achieved only by measuring exact distances between points occurring in a typical canopy after we get these canopies. Using canopies, broad clustering issues that were previously impossible are becoming feasible and efficient. An integrated method was proposed to measure the distance.

C. Distance Measure Function:

There are a number of distance metrics, but to keep this section concise, we will only be discussing a few widely used distance metrics [19]. The selection of distance measure is varied among different datasets (i.e., some measures perform better for some kind of datasets). Among all the distance measures we have adapted the best performing measure which suits for the taken dataset. Here we take 5 different distance measures and verify them against the data points and consider the distance measure which performs better by taking less time to respond which is shown below Algorithm-2.

D. Distance Measure Strategy:

Distance measurement plays a very important role in this algorithm's success. As we know, with various techniques available, the distance between two points can be computed, so our main objective is to suggest a novel technique from the available ones. However, some important points to be noted in choosing such techniques are: the data property and the dataset dimension.

Depending on the above information we adopted a new integrated distance measure for improving the performance as follows:

E. Integrated Distance Measure (IDM):

Take five distance measures as follows.

i) **City Block (Manhattan) distance:**

$$\sum_{i=1}^{k} | a_i - b_i |$$

ii) **Euclidean distance:**

$$\sqrt{\sum_{i=1}^{k} (a_i - b_i)^2}$$

iii) **Cosine Distance:**

$$d_{ab} = 1 - \frac{x_a x'_b}{\sqrt{(x_a x'_a)(x_b x'_b)}}$$

iv) **Correlation Distance**

$$d_{ab} = 1 - \frac{(x_a - \bar{x}_a)(x_b - \bar{x}_b)'}{\sqrt{(x_a - \bar{x}_a)(x_a - \bar{x}_a)'}\sqrt{(x_b - \bar{x}_b)(x_b - \bar{x}_b)'}}$$

v) **Tanimoto Distance**

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}.$$

Algorithm-2: Selection of Best Distance Measure Function:
Begin
Initialize Distance_List ← [ ]
Enumeration of distance_measures ← [Distance of City Block, Distance of Euclidean, Distance of Cosine, Distance of Correlation, Distance of Tanimoto] as 1,2,3,4 ,5
Iterate through 1,2,3,4,5 Measures
begin
Compute distance measures using 1, 2, 3, 4,5
Store all 5 distance values in Distance_List
end
Consider the one best distance measure which takes less time
End

F. The Improved model of Parallel K-means Clustering algorithm along with Improved version of Map-Reduce:

The parallel processing of the algorithm for K-means algorithm is divided into the following stages.

1) Initialization:
(a) Set of the data points for input {x_1 , x_2, x_3, ...x_n} and split the entire dataset into small sub-datasets, like split-1, split-2, split-3 and split-n. Then those the sub-datasets are formed into lists of pairs of <Key, Value>. And those <Key, Value> list are fed as input to map function.
(b) Take k-data points, obtained as clustering centroids from Canopy Clustering.

2) Mapper: Mapper
(a) When appropriate, perform updation of centroids of the clustering. And compute the distance between the centroids of the other available data points for k.
(b) Allocate each data to its closest cluster before all data gets processed.
(c) Pair <c_i, x_j> is obtained as output. And c_i is the considered as nucleus of the x_i cluster.

3) Reducer:
(a) Now read from the Map Stage <ci, xj>. Collect all the documents for results. And the k clusters as well as the data points are shown.
(b) Measure the distance with the pre-determined centroid as a canopy of every data point.
(c) Allocate that data point to the canopy based on the distance value obtained and proceed to convergence.
(d) Finally, the output of the k clusters is obtained from those data points.
The working of improved Parallel K-means algorithm is illustrated below.

Mapper Algorithm: (Centroids Calculation)
Input: Take set of objects which needs to be transformed in to a numerical vector form as
O = {or_1, or_2, …, or_n}, and consider a set of initially available cluster centers as C = {ct_1, $ct_{2...}$ ct_k}

a) F1← { or_1, or_2, …….or_n}
b) Assume the Existing Centroids ←C
c) Compute the Dist(x,y) = $\sqrt{\sum_{j=1}^{d}(x_j - y_j)^2}$, where, x & y are objects (In dimension i)
d) For the all $O_j \in$ F1 s.t $1 \leq j \leq f$ do
Initialize theBestCentroid←NULL

distance_minimum ← ∞

e) For the all values of C do this
compute the Similarity←distance(or_j,C)

f) If(theBestCenroid←null (or) similarity<distance_minimum) : then do
distance_minimum < similarity
compute C← theBestCentroid

g) End If

h) End For
release(theBestCentroid, o_i)
Increment j value by 1

i) END LOOP-FOR

j) Return intermediate <key, value> pair values

Output: The obtained intermediate <key,value>pair of (C_j,O_j)

Reducer Algorithm(Final Centroids Calculation):
Input: Take the (Key,Value) paired values obtained by the mapper. where the key = theBestCentroid and value of the objects are allocated to it.

a) compute op_pair=op_pair(<Key,Value> pairs from all mappers
b) [centroid]←{ }
c) compute theNewCentroid ←NULL
d) Assume α ← <key, value> pair is obtained from the mappers
e) Perform For all the α ϵ op_pair perform as follows
theOldCentroid ←key α (i.e., α.key)
object←α.value(i.e., α.value)
[centroid]←object (Assign)
f) End loop-For
g) For-all centroid ϵ [centroid] do perform
compute theSumofAllObjects←Null (Assign)
FOR-all the objects ϵ [centroid] : do perform
compute theSumofAllObjects ←theSumofAllObjects + object
compute theNumofObjects←theNumofObjects+1
h) End For loop

compute theNewCentroid ←(sumofObjects / theNumofObjects)
release (theOldCentroid, theNewCentroid)
i) End For

Output Obtained: <Key, Value> represents a pair, where key and values representing the old-Centroid & newCentroid respectively; thebestCentroid of mapper can used to calculate the value for newCentroid.

G. Approaches adopted to enhance the Performance:

The strategies incorporated in this improved MapReduce-based parallel K-means algorithm to improve performance as well as accuracy are outlined below[20] .

H. Adjustment of various parameters in Map & Reduce functions:

The performance of the Map & Reduce function can be improved further by adjusting various parameters in MapReduce Programming Model [21]. The following are the influencing parameters which can be

considered for improving the performance of Hadoop Clustering process as well as MapReduce functions. These are….

i) Split Size
ii) Number of Reducers
iii) Combiners
iv) Distributed Cache
v) Compression
vi) Combine Input File Format
vii) Filtering
viii) JVM Reuse

As of now, in the above list of parameters only **Split Size** and **Number of Reducers** are considered for this work [22].

i) Split Size: The block size affects sequential read and write sizes in HDFS, as well as having a direct effect on the output of the map task because of how input splits are determined by default. Here block, split and Map tasks are representing the same. The thumb rule to be considered is number of splits must be 50% of the io.sort.mb. Hence the following are the conclusions drawn for the split size adjustment.

Default block size is changed: In Hadoop 2.x, the default block size has been modified from 64 MB to 128 MB, we should know what effect the performance of a lower block size or a higher block size has and then we can determine the block size.

MapReduce coding improvements:
You can also customise the MapReduce code so that it works effectively [23] .

- **Reuse objects**: Since the map method is called several times, it can allow you to minimise overhead associated with the creation of objects by making new objects wisely. Try as best as you can to reuse artefacts. Writing the code as follows is one of the most common errors.

```
private TxtVal = new Text();
public void map(LongWritable  key, TxtVal, Context ctxt) throws IOException, InterruptedException
{
  String[] strArr = value.toString().split("\\s+");
  value.set(strArr[0]);// reusing object
  ctxt.write(key, value);
}
```

- **String concatenation:** Since String is immutable in Java, String concatenation results in the formation of String objects. StringBuffer() or StringBuilder() are chosen for appendices instead.

ii) Number of Reducers
Thumb rule is : all the reducers must be completed their job in equal amount of time.
If there is a skew, and one of the reducers use 50% of data then we have to use only 2 reducers.
If one of the reducers are using 25% of data then we have to use only 4 reducers with equal amount of time 10% of data - 10 reducers.

Note: more number of reducers → takes less time (but there are more resources)

V. EXPERIMENTAL SETUP & RESULTS AND ANALYSIS

a) Datasets:

The data sets used range from 100MB to 500MB in size. Currently, the data sets can't be called big data. They are broad enough, however, to judge the consistency of clusters. It should be noted here that HDFS

is the data source used and the size of the block is 64MB. Thus, the data sets are stored around the different nodes in the cluster.

Algorithm	**Dataset(iris)**	**Dataset(Wine)**
K-means	0.7812	0.6821
PSO+ K-means	0.9285	0.9317
Improved K-means with Improved Map Reduce **(Proposed)**	0.9462	0.9612

Table 2. Comparison of various algorithms with the algorithm proposed

b) The setup for Experimentation:

The experiment carried out initially on the Hadoop cluster of 3 nodes & subsequently increased to 10 nodes. The Hadoop cluster nodes are equipped with an Intel Core 2 Duo CPU@ 2.53 GHz cpu, 8 Giga Bytes of DDR3 RAM for every node, and 80 Giga Bytes of hard drive with a bandwidth of 100 MB/s estimated for an end-to-end sockets of TCP.

c) Cluster Configuration:

	Master	**Node01**	**Node02**	**Node03**
CPU Configuration	PC - Intel Core™, i7-4510U@2GigaHz	PC- Intel Core™, i7-4510U@2GigaHz	PC- Intel Core™, i7-4510U@2GigaHz	PC- Intel Core™, i7-4510U@2GigaHz
CPU Cores	04	02	02	02
RAM Size	8 Giga Bytes	8 Giga Bytes	8 Giga Bytes	8 Giga Bytes

Table 3. Cluster and CPU Configuration

d) Performance evaluation of K-means Algorithm with Variants:

Comparison of Single Machine and Hadoop Platform time consumption using the classical K-means algorithm [24] is shown below Table. 4.

Data Sets	**# data sets**	**K-means algorithm - Comparison With Running time(s)**	
		One Machine	**Hadoop Platform (3 Node Cluster)**
Set-1	5,000	3.8	3.0
Set-2	10,000	5.9	5.0
Set-3	15,000	7.9	7.2
Set-4	20,000	19.7	9.6
Set-5	25,000	22.3	10.4
Set-6	30,000	20.5	11.6
Set-7	35,000	30.8	14.5
Set-8	40,000	38.1	18.3
Set-9	45,000	43.2	21.4
Set-10	50,000	56.6	28.2

Table 4. K-means algorithm - Performance evaluation

From the table above.4, as compared to a single computer, it is known that the time needed for processing the data, with the Hadoop Platform is much less. And since the number of nodes in the Hadoop Cluster is increasing, the efficiency of the algorithm in the processing of the datasets is also improved, as shown in figure 5 below.

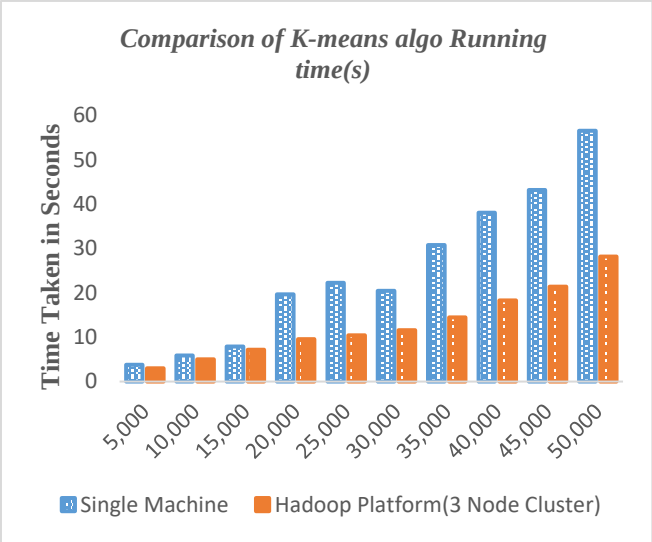

Fig.5. Comparison of the algorithm - K-means to Hadoop with Running Time (s)

e) Performance of Proposed Algorithm(Improved K-Means with Improved Map-Reduce)

The performance of the K-means clustering algorithm when processing different datasets with varied size in combination with different kinds of Hadoop cluster sizes as shown in Table. 5 below, in the MapReduce paradigm over the Hadoop architecture, as shown in Figure 6 below.

Data Set/ Node Size	**1,00,000 Points**	**1,50,000 Points**	**2,00,000 Points**	**3,00,000 Points**
3 Nodes	265	379	486	628
5 Nodes	208	288	369	474
8 Nodes	117	199	258	375
10 Nodes	62	108	127	188

Table 5. Execution Time(Seconds)

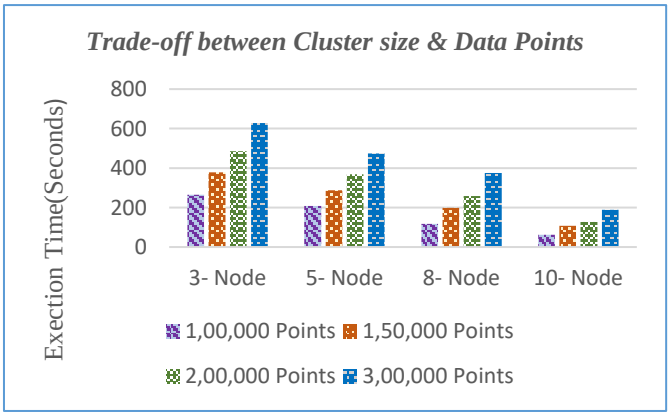

Fig 6. Execution Time

Execution Time Ratio:

Calculation of Time Ratio	Nodes =3	Nodes =5	Nodes =8	Nodes = 10
1,00,000 Points	265	208	117	62
ET Ratio	4.27	3.35	1.89	1.00
1,50,000 Points	379	288	199	108
ET Ratio	3.51	2.67	1.84	1.00
2,00,000 Points	486	369	258	127
ET Ratio	3.83	2.91	2.03	1.00
3,00,000 Points	628	474	375	188
ET Ratio	3.83	2.91	2.03	1.00

Table 6. Execution Time Ratio

In Table 6, tells us about the sequential & parallel execution of k-means with different execution times with respect to different datasets are given. To assess the efficiency of
The suggested k-means are the method of evaluation for scaling up of data. The Proposed algorithm of k-means as well as sequential k-means are conducted for data scale-up experiments with regard to different Dataset sizes are reported for each experiment with a fixed size of Ten node Hadoop clusters & execution time. The resulting experimental execution time provides a context for
Figure 7 below illustrates the study and estimation of the output differences between our proposed & sequential k-means.

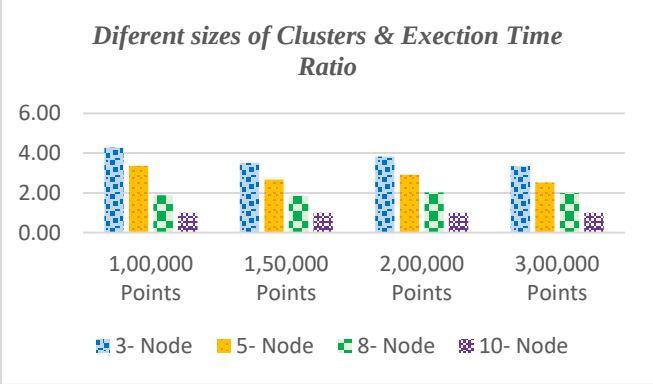

Fig 7. Execution Time ratio of different sizes of clusters

f) Comparison of algorithm performance and Validation methods used in cluster analysis:
There is a target variable in supervised learning to determine the accuracy of the model. But, in the case of unsupervised learning, how do we calculate the output of the model, because there is no target variable? The techniques for assessing the precision of clustering can be divided into two large categories:

1) Internal Measures for Accuracy and 2) External Measures for Accuracy

1) Internal Accuracy Tests: These measures determine, as the name implies, the accuracy of the cluster based on the compactness of the cluster. The methods which fall under this category are as follows:
i) Sum of Squared Errors (SSE) - By calculating its SSE, the compactness of a cluster can be calculated. When the clusters are well apart from one another, it works best. Its formula is given by (shown below) where the number of C Qu is the number of observations found in cluster and μκ is considered as mean distance in cluster k.

$$SSE = \sum_{k-1}^{K} \sum_{\forall x_i \in C_k} \|x_i - \mu_k\|^2$$

We have used the Number of Squared Errors(SSE) method to calculate the internal accuracy of our proposed algorithm, as shown below.

Error Rate & Runtime: (Power Dataset) (Power Dataset):

We used two of the UC Irvine Machine Learning Repository's wide real-world datasets. The first dataset (Power) consists of different households' energy consumption readings and comprises 512,320 real-valued data points, each with dimension 7. The second dataset (Census) consists of data from the US Census of 1990 and consists of 614,571 integer data points, each with a size of 68. And an experiment is conducted using the above datasets and the results are tabulated in Tables 7 & 8 as well as shown in Figures 8 through 13 below.

Algorithm	**Average Error**		**Min. Error**		**Average Runtime**	
	K → 5	K→ 10	K→ 5	K→ 10	K→5	K→ 10
K-Means	32.78	21.66	27.02	19.46	308	459
K-Means++	48.50	18.69	27.61	17.32	270	844
K-Means**	35.33	26.10	27.01	19.08	204	366
K-Means+*	28.06	18.35	27.01	16.59	217	465
Parallel K-Means (M-R)	26.24	16.04	27.21	15.90	210	450
Improved Parallel K-means(M-R)	24.05	15.65	27.08	13.83	198	398
Improved Parallel K-means (with Improved Map Reduce) - **Proposed**	19.48	13.42	27.75	11.24	179	345

Table 7. Error Rate and Runtime

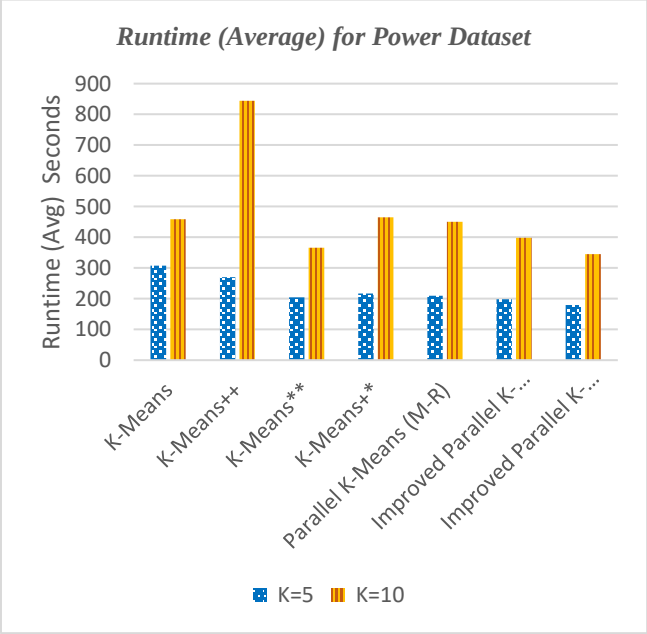

Fig 8. Runtime(Average) for power dataset

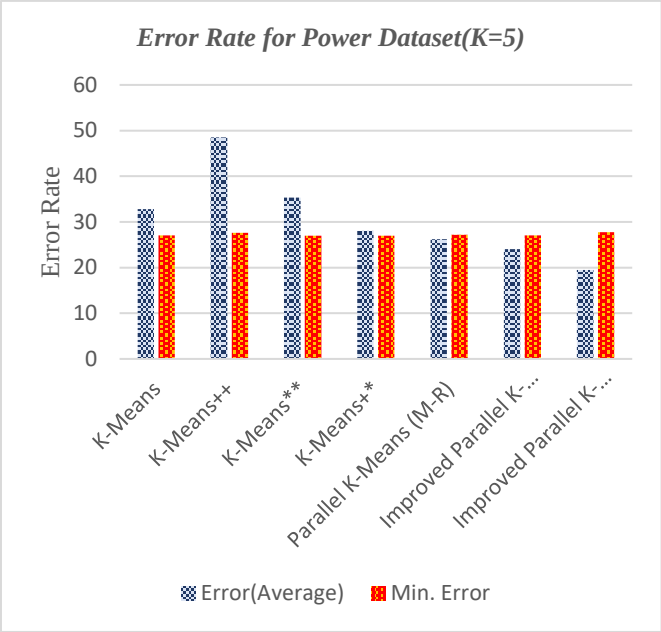

Fig 9. Error Rate for power dataset(K=5)

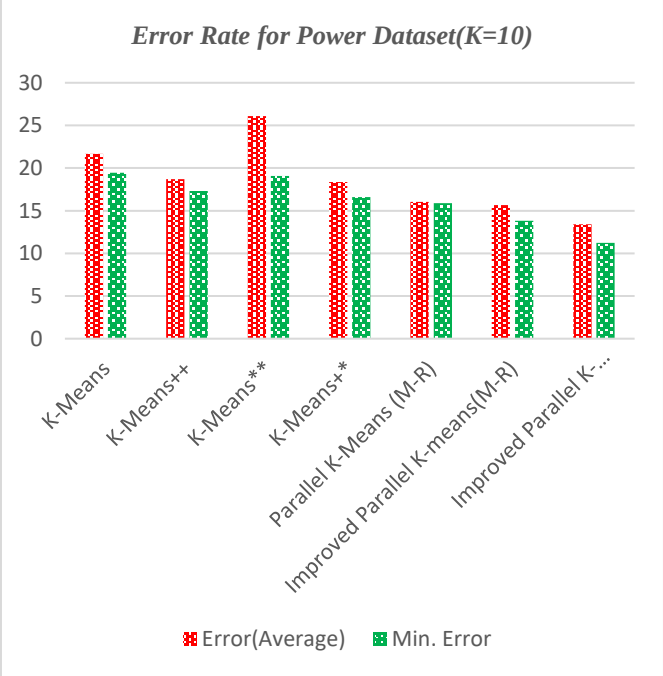

Fig 10. Error Rate for power dataset(K=10)

Error Rate & Runtime: (Census Dataset)

Algorithm	**Error (Average)**		**Min. Error**		**Runtime (Average)**	
	K→ 5	K→ 10	K→ 5	K→ 10	K→ 5	K→ 10
K-Means	397.73	154.17	136.33	100.62	402	616
K-Means++	148.44	104.05	136.33	99.64	527	1217
K-Means**	491.62	356.20	141.98	100.36	342	618

K-Means+*	153.60	105.68	136.33	99.51	396	915
Parallel K-Means (With M-R)	149.45	110.28	136.23	99.87	376	869
Improved Parallel K-means (With M-R)	138.90	99.78	139.56	99,69	298	747
Improved Parallel K-means (with Improved Map Reduce) - **Proposed**	124.21	90.63	138.28	99.98	198	568

Table 8. Error Rate and Runtime

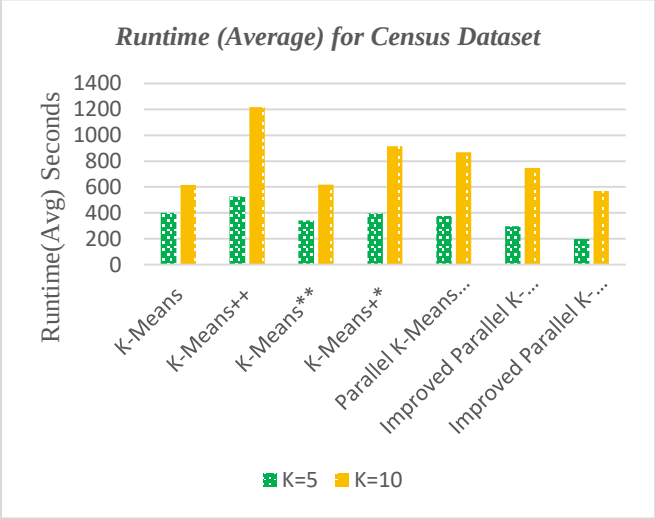

Fig 11. Runtime (Average) for Census dataset

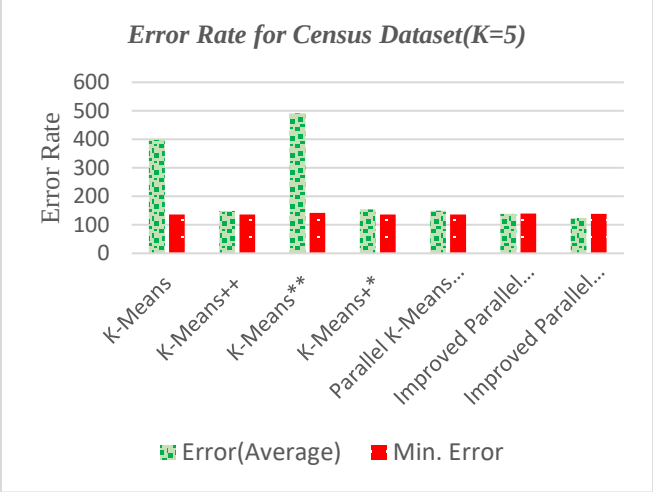

Fig 12.Error Rate for Census dataset(K=5)

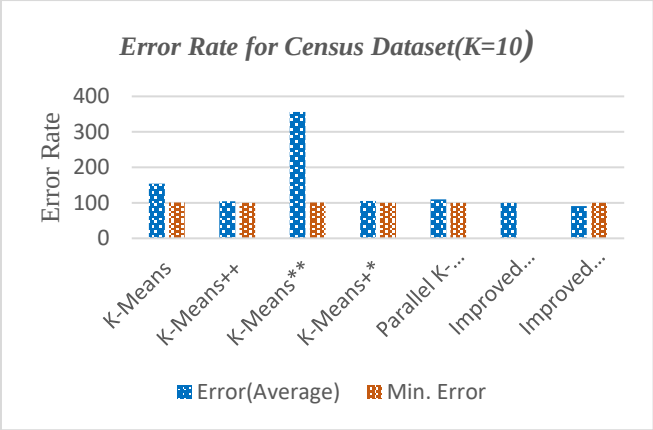

Fig 13. Error Rate for Census dataset(K=10)

(ii) Scatter Criterion - It measures the distribution of a cluster in simple terms. To do that, a scatter matrix, within cluster scatter and between cluster scatter, is determined first. It then sums in order to obtain complete scatter values over the resulting values. It is beneficial to lower values. Let's take their respective formulas to understand:
The sum of [product of (difference between the mean of the observation and its cluster) and (transpose)] is the scatter matrix. It is computed by

$$S_k = \sum_{x \in C_k} (x - \mu_k)(x - \mu_k)^T$$

The sum of all Sκ values is simply inside the cluster dispersion (Sω). It is possible to calculate the between-cluster matrix (SB) as

$$S_B = \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T$$

Where N_k denotes number of k cluster observations and the μ is the total mean vector calculated as

$$\mu = \frac{1}{m} \sum_{k=1}^{K} N_k \mu_k$$

Where m represents the number of matches within the cluster. Finally, it is possible to measure the total scatter metric S(T) as S(T) = SB +Sω

2) Measures of External Accuracy:
These measurements are determined by comparing the configuration of the clusters with certain pre-defined classifications of data instances. Let's look at steps like this:
i)Rand Index - Compares the two clusters and aims to find the ratio of matching and unmatched findings between the two structures of the cluster (C1 and C2). Its value varies from 0 to 1. Think of the structures for clustering (C1 and C2) with several small clusters. Think of C1 as the output of your projected cluster and C2 as the output of the real cluster. The greater the importance, the higher the ranking. His basic formula is given by:

$$\text{computer Rand Score} = \frac{(a + d)}{(a + b + c + d)}$$

Where,
a = findings that are available in both systems in the same cluster (C1 and C2)
b = findings that are available in C1 in a cluster and not in C2 in the same cluster

c = findings that are available in C2 in a cluster and not in C1 in the same cluster
d = observations that are available in C1 and C2 in separate clusters

ii)Precision-Recall Calculation - The uncertainty matrix is used to derive this metric. Sensitivity [True Positive/ (True Positive + False Negative)] is also known as recall. We use this metric from an information retrieval point of view for clustering. Here, accuracy is a calculation of objects correctly retrieved. Recall is the calculation of matching objects from all items that have been correctly retrieved.
The rate of accuracy is directed at the outcomes.
Two samples are used, one of which is a positive sample (TP) and the other is a positive sample (FP) and, as shown below, the precision rate measurement formula:

$$\text{computer the Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

The accuracy of the five algorithms are compared and we mark different outliers by making the same data set.
The accuracy test has been used for our proposed algorithm and the measures are tabulated as shown in the table below, Tab. 2, The algorithm's accuracy rate relation, the K-means algorithm's accuracy rate is still higher, but the whole is not robust enough, but the proposed algorithm got the better efficiency than that of them compared to the other algorithms.

Meanwhile, measuring computational data and running speed is the complexity of the algorithm, and the proposed algorithm depends not only on the broad data platform, but also uses hierarchical sampling data to allow mass data measurement. Therefore, the complexity of our proposed algorithm, as shown in Table 9.(b) below, is lower than that of conventional clustering algorithms. And different symbols were used to denote different algorithms as shown in Table 9.(a) below.

Symbol	**Algorithm**
M1	Standard K-means
M2	Parallel K-Means (With M-R)
M3	K-Means- Hadoop Map-Reduce (KM-HMR)
M4	K-Means- modified inter and intra clustering (KM-I^2C)
M5	Improved Parallel K-means with Improved Map Reduce (IKM-IMR) **Proposed**

Table 9.(a) Algorithm with symbols

#Outliers	**M1**	**M2**	**M3**	**M4**	**M5**
200	0.942	0.864	0.912	0.922	0.953
400	0.917	0.873	0.907	0.927	0.948
600	0.876	0.901	0.876	0.896	0.931
800	0.903	0.881	0.903	0.903	0.925

Table 9.(b) Time Complexity Comparison of various algorithms with the proposed algorithm.

The Precision Comparison or accuracy of different algorithms with proposed algorithm is also experimented and shown in figure 15 which reveals the accuracy of the algorithm proposed is found to be good in all aspects.

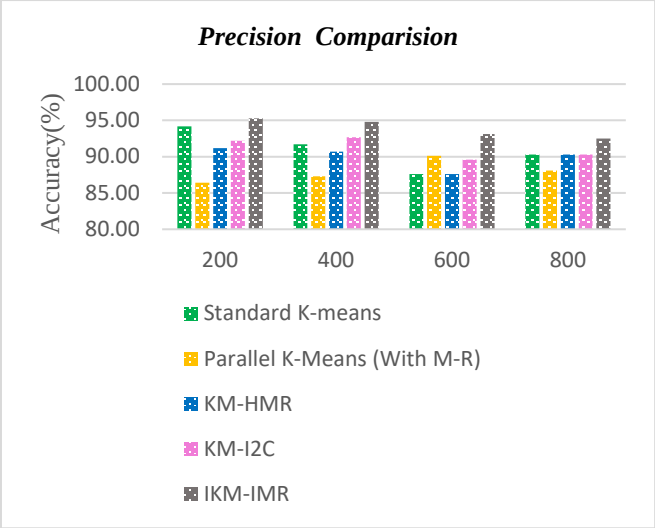

Fig 14. Precision Comparison of different algorithm with proposed algorithm.

Comparison of Algorithm Runtime:

[25] When applied to the USCRN dataset, it compares the performance of the proposed model to that of the KM-HMR and KM-I2C algorithms. Finally, our proposed algorithm, Improved Parallel K-means, appears to give us the best of the two worlds with Improved Map Reduce(IKM-IMR). It ranks first or second in each case by average error, and first in all cases by minimum error.

However, in each case, it also ranks second or third by runtime, and it is significantly faster in all cases than other existing algorithms shown in Table 10. below.

In terms of average and marginal error, its closest rival. And finally, the obtained execution time of the proposed algorithm is also contrasted with its counter parts and shown in figure 15.

Execution Period(Seconds):

Nodes	**M1**	**M2**	**M3**	**M4**	**M5**
3	4502.18	4320.14	4214.16	3876.23	3286.25
5	4025.06	3880.23	3705.29	3518.34	3118.23
8	3907.12	3691.18	3519.12	3412.49	2810.24
Avg.	4144.78	3963.85	3812.85	3602.35	3071.57

Table 10.(b) Comparison of time complexity with the proposed algorithm for various algorithms.

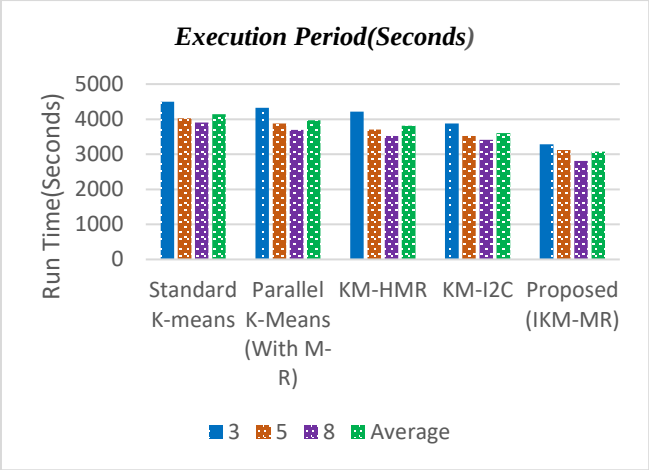

Fig 15. Duration of execution Comparison of various algorithms with the algorithm proposed.

The results obtained by running our proposed Improved Parallel K-means algorithm with the quad core machine Improved Map Reduce(IKM-IMR) are being tested. In the case where 10 nodes were used, the highest result was achieved, four times faster processing of the dataset than in the case of a 3-node cluster.

f) Execution Period(Seconds) for Microarray Data:

Experimental studies discussed here show the efficacy of the proposed study of the cluster of microarray data.[26] The five microarray gene databases used in the experiment are illustrated below, where the expression pattern of one gene is shown in each row in the data sets, and each column is an experimental sample. There have been studies on a wide range of different types of microarray data (cancer data), a few of which are summarized below. There are two groups of samples in each dataset, one category being normal and the other cancerous.

(i) Data from the GDS2771 series: 2000 rows (genes) and 72 columns exist (samples). 36 samples are natural among these, and others are considered cancerous.
(ii) GSE14407 series data: There are 54675 lines of data (genes) and 24 samples included (12 epithelial ovarian surface cells and 12 laser cells capture micro-unselected serous papillary ovarian cancers).
(iii) Data from the GSE16415 series: 32878 rows data (genes) and 10 specimens containing (5 are diabetic and 5 are control of women samples).
(iv) Carcinoma Normal Cancer Research Dataset (CNCR): This is a gene data base that is well understood. 7457 rows (genes) and 36 samples are included (18 samples are normal and others are cancerous).
(v) Adenomas Standard Cancer Research (ANCR) data: This is a well-understood database of gene data. The 7086 lines (genes) and 8 samples used are (4 samples are normal and others are cancerous). Initially, the proposed approach produces large numbers of clusters by applying the k-means algorithm with k across the range[40-60] and states that the final results have not modified significantly (in terms of validity index measure).By considering k = 50 initial cluster numbers, the effects of the clustering stated here are achieved. The final number of clusters mentioned in Table 11 is obtained after successive iterations of the merging and subsequent splitting processes. For data sets, the k-means algorithm is implemented, taking into account the same number of clusters as the k value, and a comparison between the proposed algorithm and the existing algorithms is made in Table 11. And finally, as shown in the table 11, the proposed technique was applied to Microarray data to assess the execution speed. Table 11 and Fig. 16 are shown below.

Datasets	Number of Samples(S) and Attributes(A)		Standard K-means	Parallel KMeans (With M-R)	Improved Parallel K-means with Improved Map Reduce (IKM-IMR) **Proposed**
	S	A			
GDS2771 series data	178	24481	5502.04	4001.21	2172.0
GSE14407 series data	162	2000	5136.13	4241.23	2917.33
GSE16415 series data	203	12600	4906.32	3405.32	2721.22
CNCR	253	15155	4144.78	3924.27	2160.25
ANCR	134	12601	4184.67	3954.53	2254.32

Table 11. Comparison of Time complexity of different algorithms with proposed algorithm for Microarray Data

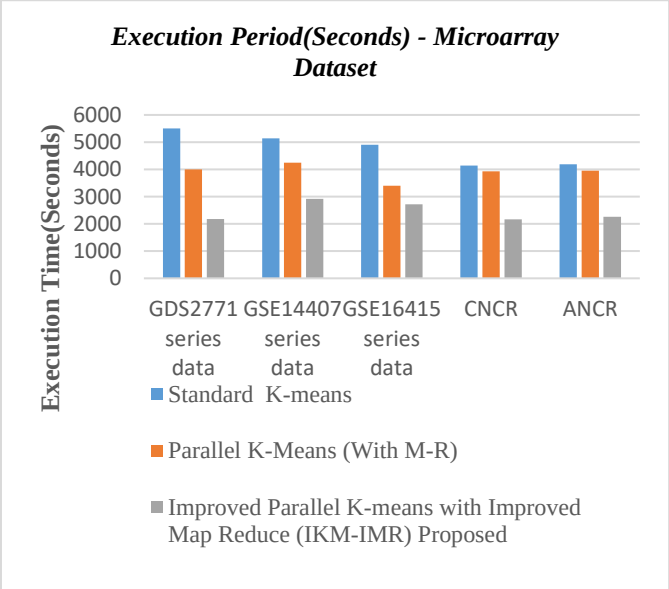

Fig 16. Execution Period Comparison of different algorithm with proposed algorithm for Microarray Data

The results obtained by running our proposed algorithm Improved Parallel K-means with Improved Map-Reduce(IKM-IMR) reveals that the best performance was obtained in processing the microarray dataset 2 times faster than in the case of Normal simple Parallel K-means approach as shown above.

VI. CONCLUSION

The improved k-means(iK-means) with improved Map Reduce(iMap-Reduce) is proposed in this paper. The combination of these two is very much recommended to increase the efficiency of the cluster of massive data produced using Canopy clustering based on the best calculation of similarity. In terms of execution time and precision, the suggested approach has shown improvements. The Map Reduce programming model Hadoop framework was used to scale up large datasets to obtain high-quality, productive clusters. By making improvements to the current clustering algorithms, we have built a novel enhanced K-Means with Improved Map-Reduce algorithm and developed an effective and more powerful

method compared to other clustering techniques and the result obtained is shown. By adapting the Mahout-based data model for scalable machine learning algorithms for efficient processing of large datasets, future work will increase map performance and decrease jobs for processing large datasets.

REFERENCES

[1] P. Shrivastava, L. Sahoo, M. Pandey, and S. Agrawal, AKM—Augmentation of K-means clustering algorithm for big data, vol. 695. Springer Singapore, 2018.
[2] M. Bodoia, "MapReduce Algorithms for k-means Clustering," Stanford.Edu, pp. 1–11.
[3] U. Bagde and P. Tripathi, "An Analytic Survey on MapReduce based K-Means and its Hybrid Clustering Algorithms," 2018 Second Int. Conf. Comput. Methodol. Commun., no. Iccmc, pp. 32–36, 2018.
[4] G. Zhou, "Improved optimization of canopy-kmeans clustering algorithm based on hadoop platform," ACM Int. Conf. Proceeding Ser., 2018.
[5] S. Y. Huang and B. Zhang, "Research on improved k-means clustering algorithm based on hadoop platform," Proc. - 2019 Int. Conf. Mach. Learn. Big Data Bus. Intell. MLBDBI 2019, pp. 301–303, 2019.
[6] P. Maniriho and A. Effendi, "Examining the Performance of K-Means Clustering Algorithm," no. March, 2018.
[7] T. Fan, "Research and implementation of user clustering based on MapReduce in multimedia big data," Multimed. Tools Appl., vol. 77, no. 8, pp. 10017–10031, 2018.
[8] C. Manike, A. K. Nanda, and T. Gajulagudem, "Hadoop Scalability and Performance Testing in Homogeneous Clusters," Lect. Notes Electr. Eng., vol. 605, pp. 907–917, 2020.
[9] M. Khan, Z. Huang, M. Li, G. A. Taylor, P. M. Ashton, and M. Khan, "Optimizing Hadoop Performance for Big Data Analytics in Smart Grid," Math. Probl. Eng., vol. 2017, 2017.
[10] S. Lee, J. Y. Jo, and Y. Kim, "Hadoop performance analysis model with deep data locality," Inf., vol. 10, no. 7, 2019.
[11] M. N. S. Zainudin, N. Sulaiman, and N. Mustapha, "Comparison of Expectation Maximization and K-means Clustering Algorithms with Ensemble Classifier Model 1 Introduction," vol. 17, pp. 253–259, 2018.
[12] K. K. Pandey and D. Shukla, "A study of clustering taxonomy for big data mining with optimized clustering mapreduce model," Int. J. Emerg. Technol., vol. 10, no. 2, pp. 226–234, 2019.
[13] I. D. Borlea, R. E. Precup, F. Dragan, and A. B. Borlea, "Parallel implementation of k-means algorithm using mapreduce approach," SACI 2018 - IEEE 12th Int. Symp. Appl. Comput. Intell. Informatics, Proc., no. 4, pp. 75–80, 2018.
[14] Y. Guo, "An Improved Parallelization of K-means Algorithm based on HADOOP," J. Phys. Conf. Ser., vol. 1187, no. 4, 2019.
[15] W. Lu, "Improved K-Means Clustering Algorithm for Big Data Mining under Hadoop Parallel Framework," J. Grid Comput., vol. 18, no. 2, pp. 239–250, 2020.
[16] A. L. Ramdani and H. B. Firmansyah, "Pillar K-Means Clustering Algorithm Using MapReduce Framework," IOP Conf. Ser. Earth Environ. Sci., vol. 258, no. 1, 2019.
[17] B. Xiao, Z. Wang, Q. Liu, and X. Liu, "SMK-means: An improved mini batch k-means algorithm based on mapreduce with big data," Comput. Mater. Contin., vol. 56, no. 3, pp. 365–379, 2018.
[18] K. Djouzi and K. Beghdad-Bey, "A Review of Clustering Algorithms for Big Data," Proc. - ICNAS 2019 4th Int. Conf. Netw. Adv. Syst., pp. 1–6, 2019.
[19] K. K. Pandey and D. Shukla, "An empirical perusal of distance measures for clustering with big data mining," Int. J. Eng. Adv. Technol., vol. 8, no. 6, pp. 606–616, 2019.
[20] A. Gandomi, A. Movaghar, M. Reshadi, and A. Khademzadeh, "Designing a MapReduce performance model in distributed heterogeneous platforms based on benchmarking approach," J. Supercomput., vol. 76, no. 9, pp. 7177–7203, 2020.
[21] M. C. M. Oo and T. Thein, "Hyperparameters optimization in scalable random forest for big data

analytics," 2019 IEEE 4th Int. Conf. Comput. Commun. Syst. ICCCS 2019, pp. 125–129, 2019.
[22] J. Morán, C. de la Riva, and J. Tuya, "Testing MapReduce programs: A systematic mapping study," J. Softw. Evol. Process, vol. 31, no. 3, pp. 1–29, 2019.
[23] N. Tsapanos, A. Tefas, N. Nikolaidis, and I. Pitas, "Efficient MapReduce Kernel k-Means for big data clustering," ACM Int. Conf. Proceeding Ser., vol. 18-20-May-, 2016.
[24] P. Wei, F. He, L. Li, C. Shang, and J. Li, "Research on large data set clustering method based on MapReduce," Neural Comput. Appl., vol. 32, no. 1, pp. 93–99, 2020.
[25] C. Sreedhar, N. Kasiviswanath, and P. Chenna Reddy, "Clustering large datasets using K-means modified inter and intra clustering (KM-I2C) in Hadoop," J. Big Data, vol. 4, no. 1, 2017.
[26] E. Gothai and K. E. College, "MapReduce based Classification for Microarray data using Parallel Genetic Algorithm Abstract :," vol. 12, no. November, pp. 4860–4866, 2016.