Parallel Semi-Supervised Big Data Clustering Based on Mapreduce Technology

Amsaveni M, Duraisamy S

Abstract: ***In the area of information technology, a speedy sensational technology is big data. Big data brings tremendous challenges to extract valuable hidden knowledge. Data mining techniques can be used over big data to extract valuable knowledge for decision making. Big data results in high heterogeneity because it consists of various inter-related kinds of objects such as audios, texts, and images. In addition to this, the inter-related kinds of objects carry different information. So, in this paper clustering techniques are introduced to separate objects into several clusters. It also reduces the computational complexity of classifiers. A Possibilistic c-Means (PCM) algorithm was introduced to group the objects in big data. PCM replicated the characteristic of each object to different clusters effectively and it had capability to avoid the corruption of noise in the clustering process. However, PCM is not more efficient for big data and it cannot confine the complex correlation over multiple modalities of the heterogeneous data objects. So, a Parallel Semi-supervised Multi-Ant Colonies Clustering (PSMACC) is introduced for big data clustering. Initially, the PSMACC splits the data into number of partitions and each partition is processed in mappers. Each mapper generates a diverse collection of three clustering components using the semi-supervised ant colony clustering algorithm with various moving speeds. Then, a hyper graph model was used to combine three clustering components. Finally, two constraints such as Must-Link (ML) and Cannot-Link (CL) are included to form a consensus clustering. Finally, the intermediate results of each mapper are combined in the reducer. However, the overhead of iteration in PSMACC is overwhelming which affects the performance of PSMACC. So, a Parallel Semi-supervised Multi-Imperialist Competitive Algorithm (PSMICA) is proposed to cluster the big data. In PSMICA, each mapper processes the ICA where initial population is called countries. Some of the best countries in the population chosen as the imperialists and the remaining countries form the colonies of these imperialists. The colonies move towards the imperialists based on the distance between them. The intermediate results of each mapper are combined in reducer to get the final clustering result.***

Keywords : Big data clustering, Parallel Semi-supervised Multi-Imperialist Competitive Algorithm, Parallel Semi-supervised Multi-Ant Colonies Clustering, Possibilistic C Means.

Manuscript published on November 30, 2019.
* Correspondence Author

Amsaveni M*, Department of Computer Science, AVP College of Arts and Science, Tirupur, India. Email: amsaveniphd2019@gmail.com

Duraisamy S, Department of Computer Science, Chikkanna Government Arts College, Tirupur, India.

I. INTRODUCTION

During the last decade, the explosion of big data [1] is high which change the way the companies gather, accumulate and examine their data to find the useful knowledge of data. This huge volume of data needs high memory requirement and it is very complex to analysis the data. It is obvious that the conventional analyzes have not kept rapidity with the growth of this change and must

progress towards greater intelligence. Clustering has been used to overcome analysis problems and it is assumed as the key of the big data analytics because the clustering process transforms the data in a condensed format that is still an information version of the whole data.

Data clustering [2] partition the data objects into several groups called clusters. The similar data objects are in the same cluster whereas the data objects in different clusters are dissimilar. Generally, clustering algorithms are classified as hard clustering and soft clustering. In hard clustering, each objects either belongs to a cluster completely or not. In soft clustering, a likelihood or probability of that object to be in those clusters is assigned. Distributed systems constantly generate a huge volume of data, which requires an extreme communication burden when all data are transferred to a central node for processing. However, in the huge volume of data the efficiency of classical soft and hard clustering algorithms is poor in terms of computation time.

Because of high computational cost, the conventional single-machine algorithms cannot handle huge volume of data. To cluster big data with high speed and scalability, multiple machine clustering techniques is used. Distributed clustering is a clustering technique which reduces communication the communication load of a mass of data across multiple machines. PCM algorithm [3] was introduced for distributed big data clustering. PCM assigned each object into multiple groups. Based on the distance between the object and the clustering center, PCM clusters the big data. It was implemented MapReduce framework to handle the big data in a distributed environment. However, it is not suitable for big data and it cannot detain the complex connection between multiple modalities of the heterogeneous data objects. So in this paper, a distributed PSMACC [4] is introduced based on MapReduce to perform big data clustering. The big data is split into number of partitions and it is processed in mappers of MapReduce. Three processes are carried out in each mapper. Initially, a diverse collection of three clustering elements are generated through the semi-supervised ant colony clustering algorithm.

These three clustering components are combined by using hyper graph model. Then, the ML and CL constraints are introduced to form a consensus clustering. The intermediate clustering results of each mapper are combined in the reducer. However, the overhead of iteration in PSMACC is overwhelming which affects the clustering efficiency of PSMACC. So, a PSMICA is proposed to cluster the big data in distributed environment.

In each mapper, PSMICA forms countries, imperialists and colonies. Based on the distance between the data objects, the colonies migrate to the imperialists to obtain the best clustering. Finally, the big data clustering result is obtained by combing the results of each mapper in the reducer.

II. LITERATURE SURVEY

A clustering Visual Assessment of Tendency (cluVAT) [5] algorithm was proposed for big data clustering. The cluVAT algorithm was used for fast clustering in big data which found roots in VAT algorithm. It re-arranged the input distance matrix to obtain the clustered data using a modified Prim's algorithm. However, it is not more suitable for small datasets.

A hybrid model called Hybrid Evolutionary algorithm with Empty Clustering ($H(EC)^2S$) [6] was proposed for big data clustering. $H(EC)^2S$ chosen representative points to remove empty clustering problem. After that, the hybrid algorithm utilized only the representative points during the centroids selection. The $H(EC)^2S$ model combined cuckoo-search and fireworks based evolutionary algorithm with some centroids-calculation heuristics for big data clustering. However, the running time of this model is high.

A parallel K-Medoids algorithm [7] was proposed for big data clustering. K-Medoids processed and assigned n number of data points into k clusters with k medoids. It tried to maintain the lowest inter-cluster similarity and highest intra-cluster similarity between the clusters. In the parallel K-Medoids algorithm, the data was split into number of partitions. Each partition was process in individual mappers where K-Medoids process was carried to cluster the data. The intermediate results of each mapper were merged in the mergers to get the final clustering result. However, parallel K-Medoids algorithm was not applicable to clusters of large scale data intensive applications.

A fuzzy c-means and cluster ensemble with random projection method [8] was proposed for big data clustering. It was an ensemble method performed based on partition on similarity graph. A new data was created for each random projection process and a fuzzy c-means clustering was applied on the new data. It returned a membership matrices and its elements were processed as similarity measures between cluster centers and data points. By applying Singular Value Decomposition (SVD) on the concatenation of membership matrices, spectral data points were obtained. However, there is no proper explanation about how to select proper number of random projections for cluster ensemble method.

A modified K-means algorithm [9] was proposed to cluster the big data. The time consumption problem due to the improper selection k values in K-means algorithm was solved by modified K-means algorithm. It found initial centroids and generated an interval between those data elements which will not alter their clusters in the succeeding iterations. Based on the distance between the centroids and data points, big data was clustered. The modified K-means algorithm minimized the workload significantly in case of big datasets. However, it required high execution time to find the initial centroids.

A new distributed clustering approach [10] was proposed to cluster big data. This approach was combination of local result generation process and global model generation process. In the local result generation process, the datasets were analyzed using Density Based Spatial Clustering Application with Noise (DBSCAN) and K-means algorithms. Then in the global model generation process, an aggregation phase was designed where the results of local result generation process were aggregated in such a way that the final cluster was compact and accurate. However, the quality of clustering heavily depends on the local clustering results.

Based on tensor canonical polyadic decomposition, an efficient Fuzzy C-means (FCM) approach [11] was proposed for big data clustering. In this approach, the conventional FCM clustering was transformed into tensor format by using a bijection function in order that the canonical polyadic decomposition compressed the attributes. In order to minimize the attributes of every object, the tensor canonical polyadic decomposition was used. The FCM was expanded to a high-order FCM method to make the clustering operation executed on the compressed objects in the tensor space. Sometimes, the initialization affects the efficiency of FCM.

A novel approach based on Interval Type-2 fuzzy uncertainty modeling was proposed [12] to improve the clustering results in big data. A gene expression data was collected as matrix and it was transformed into interval type-2 fuzzified data through a membership function generation process. After that, a crisp corresponding of the fuzzified dataset was obtained by applying an improved Nie-Tan defuzzification method. The defuzzified data were clustered using FCM clustering. However, it does not work well with large datasets.

III. PROPOSED METHODOLOGY

In this section the proposed PSMACC and PSMICA are described in detail for big data clustering. Initially, the attributes of big data is reduced by Parallel Rough set Theory-based Attribute Reduction (PRT-AR) [13]. Then, the PSMACC and PSMICA are designed based on MapReduce to perform big data clustering in a distributed environment. MapReduce consists of map function and reduce function. Map function automatically split the big data into number of partitions. Then, the semi-supervised multi-ant colonies clustering and semi-supervised multi-Imperialist Competitive Algorithm are processed in each map function for big data clustering. It also removes the irrelevant data in the dataset. In the reduce function, the result of each map function is combined to return a final clustering result.

A. Parallel Semi-Supervised Multi-Ant Colonies Clustering for Big Data Clustering

The Semi-supervised Multi-Ant Colonies Clustering (SMACC) algorithm is the basic of the PSMACC. In the SMACC, data objects are randomly anticipated onto a plane with a Cartesian grid.

Then, each ant chooses a data object at random. Each ant pick up or moves or drop down the data object, based on the dropping or picking-up probability of the present data object within a local region.

The semi-supervised multi-ant colonies clustering algorithm is started with initializing the population of ants in each map function. Consider, an ant is situated at place p and finds a data object obj_i at that place. The local density of data objects similar to obj_i at p is given as follows:

$$f(obj_i) = \max\left\{0, \frac{1}{h^2} \sum_{obj_i \in N_{h\times h(p)}} \left[1 - \frac{d(obj_i - obj_j)}{\sigma\left(1 + ((q-1)/q_{max})\right)}\right]\right\} \quad (1)$$

In Eq. (1), $f(obj_i)$ is a average similarity density measure of data object obj_i with other data object obj_j present in its neighborhood. $N_{h\times h(p)}$ denotes a square of $h \times h$ places surrounding place p. $d(obj_i - obj_j)$ is the cosine distance between obj_i and obj_j in the space of attributes. A factor that describes the level of similarity between data objects is represented as σ.

The speed of the ants is denoted as q and the maximum speed of the ants is denoted as q_{max}.The ants which are moving at faster speed make clusters approximately on large scales whereas the ants are moving at slow speed cluster data objects at smaller scales by keeping objects with more accuracy. According to the moving speeds of ants three types of clustering elements are formed. Three different moving speeds of ants are listed as follows:

1. q is random. The speed of each ant is distributed randomly in $[1, q_{max}]$.
2. q is constant. At any time all the ants in the population moves at the same speed.
3. q is arbitrarily decreasing. The speed term begins with a huge value, then the speed value is steadily decreased in a random manner.

The probability conversion function is a function of $f(obj_i)$. It converts the mean similarity of a data object into the probability of dropping-down or picking-up for an ant. The dropping-down probability P_{drop} for a randomly moving loaded ant to deposit an object is given as follows:

$$P_{drop} = sigmoid\left(f(obj_i)\right) \quad (2)$$

The picking-up probability for a randomly moving ant is given as follows,

$$P_{pick} = 1 - sigmoid\left(f(obj_i)\right) \quad (3)$$

In Eq. (3), $sigmoid()$ is the sigmoid function. To guide the clustering process towards an accurate grouping, pair wise constraints are used in PSMACC. It is noted that the higher the similarity of a data object is, the higher dropping-down probability is and conversely. So, ML and CL constraints are used to decide whether the ants picking-up or dropping-down the objects. The CL denotes that a pair of data cannot be in a cluster and ML requires that two data objects must be in the same cluster.

Consider, N_{ML} is the number of ML constraints among the object obj_i situated at place p and the other objects obj_j presenting in $N_{h\times h(p)}$ of the object obj_i that can be given as follows:

$$N_{ML} = \begin{cases} N_{ML} + 1 & if\ (obj_i, obj_j) \in ML \\ 0 & otherwise \end{cases} \quad (4)$$

Consider, N_{CL} is the count of CL constraints among the data object obj_i suited at place p and the other data object obj_j is there in $N_{h\times h(p)}$ of the object obj_i that can be given as follows:

$$N_{CL} = \begin{cases} N_{CL} + 1 & if\ (obj_i, obj_j) \in CL \\ 0 & otherwise \end{cases} \quad (5)$$

When N_{ML} is greater than a user specified $cons$, it implies there are many data objects that must belong to the same cluster in this data object's neighborhood so that the ant drop down the data object. If N_{CL} is greater than $cons$, then it denotes the data object is unlikely to its neighborhood. Therefore, the ant must pick it up and migrate it to a new position.

In order to ensemble the clustering results of different clustering components, a hyper graph model is used. Consider a set of data objects as $obj = \{obj_1, obj_2, \ldots obj_n\}$ and clustering elements of these n data objects into k clusters can be denoted as a label vector $\tau \in N^n$. For y group clustering components with the m-th grouping $\tau^{(m)}$ having $k^{(m)}$ clusters, a binary indicator matrix $H^{(m)\in I^{n\times k^{(m)}}}$ is built, in which each cluster is denoted as a hyperedge. A concatenated block matrix is further defined as the adjacency matrix of a hypergraph with n vertices and $\sum_{m=1}^{r} k^{(m)}$ hyperedges.

$$H = \left(H^{(1)}, \ldots, H^{(r)}\right) \quad (6)$$

In matrix H, each row and column represents a data object and hyperedge respectively. In the hyperedge, 0 represents that it is not or the data is unknown and 1 represents that the vertex matching to the row fit in to the same cluster. Thus, the clustering elements are transformed into a hypergraph with the adjacency matrix H. Eq. (7) represents the aggregated similarity matrix M among n objects.

$$M = \frac{1}{r} H H^T \quad (7)$$

In Eq. (7), H^T is the transposition matrix of H and M is $n \times n$ sparse matrix. For ML and CL constraints are frequently denoted as:

$M_{ij} = 0$, if (obj_i, obj_j) is unlikely to be in the same class.

$M_{ij} = 1$, if (obj_i, obj_j) is likely to be in the same class.

These are used to update the value of M which enhance the clustering ensemble accuracy.

The PSMACC is the implementation of semi-supervised multi-ant colonies clustering algorithm in MapReduce. Map automatically splits the big object set $Obj = (Obj_1, Obj_2, \dots Obj_n)$ into n number of partitions and then each map function reads the content of the corresponding data split. Obj is randomly anticipated onto a plane, and each data object in obj_i is noticeable by a two-dimensional coordinate $C(obj_i)$.

By assigning the location of every ant $C(l_i)$ to be (0,0) and the loading state $State(l_i)$ to be unloaded, the ant colony is initialized. Then, side length of local region (s), number of ants (N_{ant}), number of must-link constraint (N_{ML}), maximum number of iteration (max_{itr}) and the number of cannot-link constraint (N_{CL}) are initialized.

Every ant in the population, the processor calculates the number of places around it in the given region s, and determines the frequency of ML and CL constraints. The ant l_i is decided to dropped down or picked up data object based on the N_{ML} and N_{CL}. If both ML and CL constraints fail to make a decision, then ant calculates the similarity of data object within a local region s by using equation (1). The map function of MapReduce returns a key-value pair is $\langle ID, f_k(obj_i)\rangle$, where ID is the succeeding number of obj_i and $f_k(obj_i)$ is the similarity within kth split.

When the ant is unloaded, then the picking-up probability P_{pick} is computed. On the other hand, if the ant is loaded, then the dropping probability P_{drop} is calculated. Based on the picking-up probability and dropping probability, the ant selects a data object. This process is carried out in the map function. Then, the global mean similarity from outputs of map function is calculated in the reduce function which is given as follows,

$$f(Obj_i) = \frac{1}{n}\sum_{k=1}^{n} f_k(Obj_i) \qquad (8)$$

Based on the probability calculated from Eq. (2) and Eq. (3), the data object are either dropped down or picked up by the ants. The reducer updates the location $C(l_i)$ and state $State(l_i)$ of ant l_i. Then, every object Obj_i will be assembled into one specific cluster. If the data object is separated or the count of its neighbor is less than N_{ML} and N_{CL} constraints, then labeled the data object as an outlier. The parallelism of PSMACC reduces the computation time of big data clustering.

Parallel Semi-supervised Multi-Ant Colonies Clustering Algorithm

```
Initialize $N_{ant}, max_{itr}, s, N_{ML}, N_{CL}, cons$
Project object $obj$ on a plane with two-dimensional coordinate $C(obj_i)$
Initialize $C(a_i) = 0, State(a_i) = unload$
for $q = [\text{random}, \text{constant}, \text{arbitarilyDecreasing}]$
do
    while $itr < max_{itr}$
        Split $Obj$ to parallel parts as $Obj^1, Obj^2, \dots Obj^n$
        for $l_i: [l_1, l_2, \dots l_{N_{ant}}]$ do
            Unloaded $l_i$ randomly selects a data object
            Compute the number of places around it in region $s$ and count $N_{ML}$ and $N_{CL}$
            if $N_{ML} > cons$
                Ant drops down currently loaded data object
            end if
            if $N_{CL} > cons$
                Ant picks up the current data object
            end if
            Calculate the similarity in a local region
            Compute the picking-up probability $P_{pick}$ when the ant is unloaded
            if ($P_{pick} >$ $random\ probability$ && $obj_i$ $is\ not\ picked\ up\ by\ the\ other\ ants$)
                Ant tags itself as loaded and moved to a new position when that ant picks up the $obj_i$.
            else
                Ant does not pick up the $obj_i$ and reselects another object randomly
            end if
            Compute the dropping probability $P_{drop}$ when the ant is loaded
            if $(P_{drop} > random\ probability)$
                Ant drops the $obj_i$, tags itself as unloaded and reselects a new object randomly
            else
                Ant continues moving the $obj_i$ to a new position
            end if
            for $obj_i: [obj_1, obj_2, \dots obj_n]$ do
                if number of its neighbor is less than a given constant or $obj_i$ is isolated
                    Label outlier
                else
                    Give $obj_i$ a cluster sequent number and repeatedly tag the same sequent number to those objects and their neighbors within the local region
                end if
            end for
    end while
end for
Calculate $H$ of hyper graph
Compute the similarity matrix $M$
Update $M$ with $ML$ and $CL$ constraints
```

The above PSMACC algorithm clusters the big data effectively. However, the overhead of iteration in PSMACC is overwhelming which affects the performance of PSMACC.

B. Parallel Semi-Supervised Multi- Imperialist Competitive Algorithm for Big Data Clustering

The PSMICA is used to cluster big data based on the ICA. Like PSMACC, PSMICA starts with in an initial population called countries which represents the data objects in the big data.

Some of the best countries in the population are chosen as imperialists and the remaining data objects in the population form the colonies of these imperialists. Based on the imperialist's power, all the colonies of initial population are divided among the mentioned imperialists. The power of an empire is based on the fitness function. The fitness function is calculated based on the inter-cluster distance and intra-cluster distance.

After dividing all colonies among imperialists, these colonies starts migrate to their relevant imperialist country. The total power of an empire depends on both the power of the imperialist country and the power of its colonies. After that, the imperialistic competition begins among all the empires. Any empire that is not able to succeed in this competition and can't increase its power will be eliminated from the competition. The imperialistic competition will steadily result in a decrease in the power of weaker empires and an increase in the power of powerful ones.

Weak empires will lose their power and ultimately they will collapse. The movement of colonies towards their relevant imperialists along with competition among empires and also the collapse mechanism will hopefully cause all the countries to converge to a state in which there exists just one empire in the world and all the other countries are colonies of that empire. In this ideal new world colonies, have the same position and power as the imperialist.

Map automatically splits the big object set $Obj = (Obj_1, Obj_2, \dots Obj_{nsplits})$ into $nsplits$ number of partitions and then each map function reads the content of the corresponding data split. To initialize $country$ in each map function, initialize the number of empires. The population represents the big data objects. In a N dimensional big data clustering problem, a country is a $1 \times N$ array. This array is described as follows:

$$country = [obj_1, obj_2, \dots obj_N] \qquad (9)$$

The price of a $country$ is determined by calculating the cost function f at the $(obj_1, obj_2, \dots obj_N)$. Then,

$$cost_i = f(country_i) = f(obj_1, obj_2, \dots obj_{iN}) \quad (10)$$

$$f(country_i) = max\left(\frac{D_{IntraClust}(country_i)}{D_{InterClust}(country_i)}\right) \quad (11)$$

In Eq. (11), $D_{IntraClust}(country_i)$ is the intra-cluster distance of i-th country and $D_{InterClust}(country_i)$ is the inter-cluster distance of i-th country.

The PSMICA algorithm starts with initial N countries and the countries which have minimum cost N_{imp} selected as imperialists. The rest of the countries are colonies that each belong to an empire. The initial colonies belong to imperialists in convenience with their powers. The normalized cost of an imperialist is defined to proportionally distribute the colonies among imperialists. The normalized cost is given as follows:

$$C_n = max_i cost_i - cost_n \qquad (12)$$

In Eq. (12), $cost_n$ is the cost of n-th imperialist and C_n is its normalized cost. Each imperialist that has more cost value, will have less normalized cost value. The colonies distributed among the imperialist countries based on the power of each imperialist which is calculated as follows,

$$power_n = \left|\frac{cost_n}{\sum_{i=1}^{N_{imp}} cost_i}\right| \qquad (13)$$

On the other hand, the normalized power of an imperialist is assessed by its colonies. Then, the initial number of colonies of an empire will be,

$$NC_n = rand\{power_n \times (N_C)\} \qquad (14)$$

In Eq. (14), the initial number of colonies of nth empire is denoted as NC_n and the number of all colonies is denoted as N_C.

NC_n of the colonies is chosen randomly and is assigned to their imperialist for the distribution of the colonies among imperialist. The imperialist countries absorb the colonies towards themselves using the absorption policy. It makes the main core of the PSMICA and causes the countries migrate to their minimum optima. The imperialists absorb these colonies towards themselves with respect to their power which is defined as,

$$TC_n = cost(imp_n) + \alpha mean\{cost(colonies\ of\ empire_n)\} (15)$$

In Eq. (15), TC_n is the total cost of the nth empire and α is a positive number which is less than 1.

Based on TC_n, the imperialists countries started to improve their colonies. This reality is represented by moving all the colonies towards the imperialist. By a random value i.e., equivalently distributed between 0 and $\beta \times d$, a colony migrate towards the imperialist which is given as follows,

$$\{x\}_{new} = \{x\}_{old} + U(0, \beta \times d) \times \{V_1\} \qquad (16)$$

In Eq. (16), $\beta > 1$, d is the distance between colony and imperialist and $\{V_1\}$ is a vector which its start point is the previous location of the colony and its direction towards the imperialists locations.

A random amount of deviations is included to the direction of movement to increase the searching around the imperialist. The new location is obtained by deviating the previous location of the country as great as θ. θ is a random number with uniform distribution as,

$$\theta = U(-\delta, +\delta) \qquad (17)$$

While moving toward the imperialist countries, a colony may reach to a better position, so the colony position changes based on the position of the imperialist. During the imperialistic competition, the weak empires will lose their power and their colonies. The probability of possessing all the colonies is calculated to model this competition. It is calculated by each empire considering the total cost of empire.

$$NTC_n = max_i\{TC_i\} - TC_n \qquad (18)$$

In Eq. (18), TC_n is the total cost of nth empire and NTC_n is the normalized total cost of the nth empire. Having the normalized total cost the possession probability of each empire is calculated as below:

$$power_{power_n} = \left| \frac{NTC_n}{\sum_{i=1}^{N_{imp}} NTC_i} \right| \qquad (19)$$

After a particular number of iteration, all the empires except the most powerful empire will collapse and all colonies will be under the control of this unique empire. The $\langle ID, power_{power_{nk}}(Obj_i)\rangle$ key-value pair is as the output of the map procedure, where ID is the succeeding number of obj_i and $power_{power_{nk}}(Obj_i)$ is the normalized total cost of each empire in the k-th split. The reduce function calculates the global average normalized total cost from outputs of each map function. It is calculated as,

$$power_{power_n}(Obj_i) = \frac{1}{nsplit} \sum_{k=1}^{nsplit} power_{power_{nk}}(Obj_i) \qquad (20)$$

In Eq. (20), $nsplit$ is the number of splits of big data objects.

Parallel Semi-supervised Multi- Imperialist Competitive Algorithm

```
Initialize $max_{itr}$, number of empires and their colonies position as big data objects
    while $itr < max_{itr}$
        Split $Obj$ to parallel parts as $Obj^1, Obj^2, \dots Obj^n$
        for $country_i: [country_1, country_2, \dots country_N]$ do
            Calculate the cost of each country
            Choose the countries with maximum cost as imperialist and the rest of the population is called as colonies.
            Colonies move towards the imperialist position
            Compute the total cost of all empires related to the power of both imperialist and its colonies.
            Select the weakest colony or colonies from the weakest empire and give them to the empire that has the most likelihood to posses it.
            Remove the powerless empires
            If there is just one empire, then stop else continue
            end for
    for $country_i: [country_1, country_2, \dots country_N]$ do
    update $country_i$ in $Country$
    end for
    for $obj_i: [obj_1, obj_2, \dots obj_n]$ do
    If $obj_i$ is isolated, label it as outlier
    else give $obj_i$ a cluster sequent number, and repeatedly tag the same sequent number to those objects who is the empire of this object.
    end for
    end while
```

The clustered big data by PCM, PSMACC and PSMICA are given to three different classifiers are Support Vector Machine (SVM), AdaBoost and Random Forest (RF) to classify the data.

IV. RESULT AND DISCUSSION

In this section, the effectiveness of PCM, PSMACC and PSMICA for big data clustering are tested in terms of accuracy, precision, recall and computation time. Amazon customer review dataset, REUTERS-21578 text dataset and International Cancer Genome Consortium (ICGC) on AWS datasets are used for experimental purpose. Amazon customer review dataset consists of 130 million+ customer reviews from 5 different countries. The REUTERS-21578 text dataset contains 21578 Reuters news documents from 1987. The ICGC on AWS consists of data about cancer which consists of 2178 donors and 40152 files.

A. Accuracy

Accuracy is the fraction of correct classifications to the total number of instances evaluated. It can be calculated as,

$$Accuracy = \frac{True\ Positive\ (TP) + True\ Negative(TN)}{TP + False\ Positive\ (FP) + TN + False\ Negative(FN)}$$

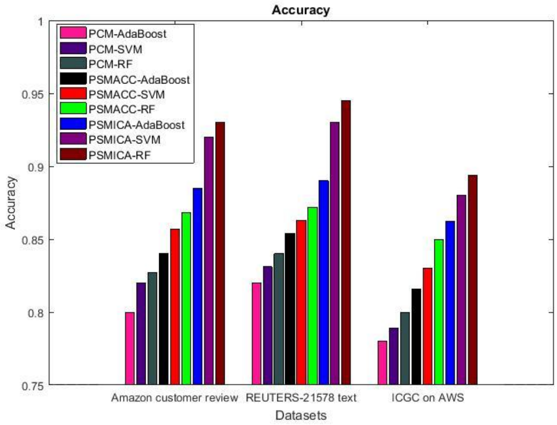

Fig.1: Evaluation of PCM, PSMACC and PSMICA using the parameter accuracy

Fig. 1 shows the accuracy of PCM, PSMACC and PSMICA with Adaboost, SVM and RF classifiers for three different datasets. The Amazon customer review, REUTERS-21578 text and ICGC on AWS datasets are taken in X axis and accuracy is taken in Y axis. The accuracy of PSMICA-RF is 1.1% greater than PSMICA-SVM, 5.1% greater than PSMICA-AdaBoost, 7.1% greater than PSMACC-RF, 8.5% greater than PSMACC-SVM, 10.7% greater than PSMACC-AdaBoost, 12.5% greater than PCM-RF, 13.4% greater than PCM-SVM and 16.3% greater than PCM-AdaBoost for Amazon customer review dataset. From this analysis and from figure 1, it is proved that the PSMICA-RF has better accuracy than the other methods.

B. Precision

Precision is used to measure the positive classes that are correctly classified from the total predicted patterns in a positive class. It can be calculated as,

$$Precision = \frac{TP}{TP + FP}$$

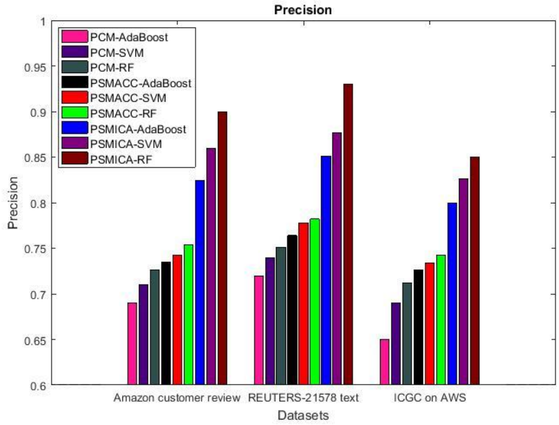

Fig.2: Evaluation of PCM, PSMACC and PSMICA using the parameter precision

Fig. 2 shows the precision of PCM, PSMACC and PSMICA with Adaboost, SVM and RF classifiers for three different datasets. The Amazon customer review, REUTERS-21578 text and ICGC on AWS datasets are taken in X axis and precision is taken in Y axis. The precision of PSMICA-RF is 4.7% greater than PSMICA-SVM, 9.2% greater than PSMICA-AdaBoost, 19.4% greater than PSMACC-RF, 21.3% greater than PSMACC-SVM, 22.4% greater than PSMACC-AdaBoost, 24% greater than PCM-RF, 26.8% greater than PCM-SVM and 30.8% greater than PCM-AdaBoost for Amazon customer review dataset. From this analysis and from figure 2, it is proved that the PSMICA-RF has better precision than the other methods.

C. Recall

Recall is used to measure the fraction of positive patterns that are correctly classified. It can be calculated as,

$$Recall = \frac{TP}{TP + TN}$$

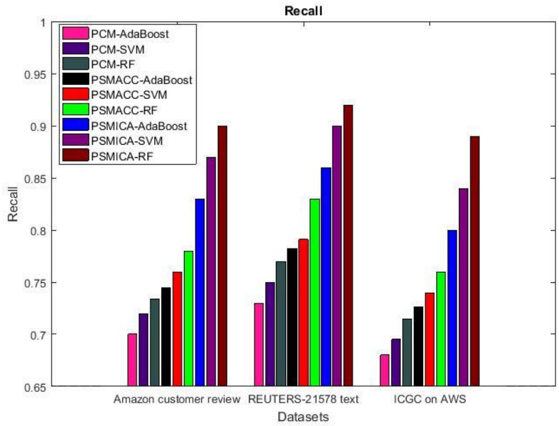

Fig.3: Evaluation of PCM, PSMACC and PSMICA using the parameter recall

Fig. 3 shows the recall of PCM, PSMACC and PSMICA with Adaboost, SVM and RF classifiers for three different datasets. The Amazon customer review, REUTERS-21578 text and ICGC on AWS datasets are taken in X axis and recall is taken in Y axis. The recall of PSMICA-RF is 3.4% greater than PSMICA-SVM, 8.4% greater than PSMICA-AdaBoost, 15.4% greater than PSMACC-RF, 18.4% greater than PSMACC-SVM, 20.8% greater than PSMACC-AdaBoost, 22.6% greater than PCM-RF, 25% greater than PCM-SVM and 28.6% greater than PCM-AdaBoost for Amazon customer review dataset. From this analysis and from figure 3, it is proved that the PSMICA-RF has better recall than the other methods.

D. Computation Time

Computation time is the amount of time taken to classify the big data.

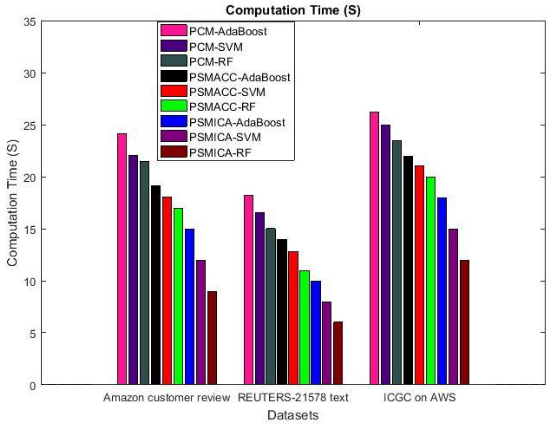

Fig.4: Evaluation of PCM, PSMACC and PSMICA using the parameter computation time

Fig. 4 shows the computation time of PCM, PSMACC and PSMICA with Adaboost, SVM and RF classifiers for three different datasets. The Amazon customer review, REUTERS-21578 text and ICGC on AWS datasets are taken in X axis and computation time in terms of seconds is taken in Y axis. The computation time of PSMICA-RF is 25% less than PSMICA-SVM, 40% less than PSMICA-AdaBoost, 47.1% less than PSMACC-RF, 50% less than PSMACC-SVM, 53% less than PSMACC-AdaBoost, 58% less than PCM-RF, 59.2% less than PCM-SVM and 62.7% less than PCM-AdaBoost for Amazon customer review dataset. From this analysis and from figure 4, it is proved that the PSMICA-RF has better computation time than the other methods.

V. CONCLUSION

In this paper, PSMACC and PSMICA are introduced for big data clustering. Initially, the attributes of big data are reduced by PRT-AR. Then, the clustering process is carried in a distributed environment based on MapReduce to remove the irrelevant data and group the similar type of data objects. The MapReduce splits the big data into number of partitions and the in each map function the PSMACC and PSMICA processes are carried out. PSMACC is based on the ant colony algorithm and the PSMICA is based on the ICA algorithm.

Finally, the reducer function updates the ant and empires based on the results of each map function. It returns a final big data clustering result. The clustered big data are given as input to AdaBoost, SVM and RF to classify the big data. The experiments are carried out in Amazon customer review, REUTERS-21578 text and ICGC on AWS datasets which prove that the proposed PSMICA is better than PSMACC and PCM methods in terms of accuracy, precision, recall and computation time.

REFERENCES

1. M. S. Hidri, M. A. Zoghlami, and R. B. Ayed, “Speeding up the large-scale consensus fuzzy clustering for handling Big Data,” *Fuzzy Sets and Syst.*, vol. 348, pp. 50-74, 2018.
2. B. Panda, S. Sahoo, and S. K. Patnaik, “A comparative study of hard and soft clustering using swarm optimization,” *Int. J. Scientific Eng. Res.*, vol. 4, pp. 785-790, 2013.
3. Q. Zhang, L. T. Yang, Z. Chen, and P. Li, “PPHOPCM: Privacy-preserving high-order possibilistic c-means algorithm for big data clustering with cloud computing,” *IEEE Trans. Big Data,* 2017.
4. Y. Yang, F. Teng, T. Li, H. Wang, H. Wang, and Q. Zhang, “Parallel semi-supervised multi-ant colonies clustering ensemble based on mapreduce methodology,’ *IEEE Trans. Cloud Comput.*, vol. 6, pp. 1-12, 2018.
5. D. Kumar, J. C. Bezdek, M. Palaniswami, S. Rajasegarar, C. Leckie, and T. C. Havens, “A hybrid approach to clustering in big data,” *IEEE trans. cybern.*, vol. 46, pp. 2372-2385, 2015.
6. Karimov, J., & Ozbayoglu, M. (2015, October). High quality clustering of big data and solving empty-clustering problem with an evolutionary hybrid algorithm. In *2015 IEEE International Conference on Big Data (Big Data)* (pp. 1473-1478). IEEE.
7. M. O. Shafiq, and E. Torunski, “A parallel K-medoids algorithm for clustering based on MapReduc,” *15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, pp. 502-507, 2016.
8. M. Ye, W. Liu, J. Wei, and X. Hu, “Fuzzy-means and cluster ensemble with random projection for big data clustering,” *Math. Probl. Eng., 2016*.
9. S. A. Fahad, and M. M. Alam, “A modified K-means algorithm for big data clustering,” *Int. J. Sci., Eng. Comput. Technol.*, vol. 6, pp. 129-132, 2016.
10. M. Bendechache, M. T. Kechadi, and N. A. Le-Khac, “Efficient large scale clustering based on data partitioning,” *IEEE Int. Conf. Data Sci. Adv. Anal. (DSAA)*, pp. 612-621, 2016.
11. F. Bu, “An efficient fuzzy c-means approach based on canonical polyadic decomposition for clustering big data in IoT,” *Futur. Gener. Comput. Syst.*, 2018.
12. A. K. Shukla, and P. K. Muhuri, “Big-data clustering with interval type-2 fuzzy uncertainty modeling in gene expression datasets,” *Eng. Appl. Artif. Intell.*, vol. 77, pp. 268-282, 2019.
13. M. Amsaveni, and S. Duraisamy, “A parallel rough set theory for non-linear-reduction in big data analysis,” *Int. J. Intel. Eng. Syst.*, vol. 12, pp. 170-178, 2019.

AUTHORS PROFILE

Ms. M. Amsaveni has around 4 years of teaching experience. She is currently an Assistant Professor at AVP College of Arts and Science College, Tirupur Department of Computer Science. She has published 3 research articles in International Journals. She has attended both national and international conferences. Her research interests include Data Mining and Cloud Computing.

Dr. S. Duraisamy received Ph.D in Computer Science in 2008. He has around 21 years of teaching experience. He is currently an Associate Professor at PG & Research Department of Computer Science, Chikkanna Govt Arts College, Tirupur. He has published more than 30 research articles in International Journals. He has acted as resource person in both national and international conferences. He has Life time Member in ISTE from 2009 onwards. He has also organized Seminars/ Conferences/Workshops organized by various funding agency such as AICTE, DRDO and TCS. His research interests include Operating Systems, Software Quality Assessment, Software Quality Management, Object Oriented Software Engineering, Java Programming, Data Base Management Systems, Data Structure Computer Networks, Routing Algorithms, and Cloud Computing. He has guided (Ph.D) more than 9 students from Bharathiar University and Anna University till now.