EasyChair Preprint

№ 1931

Optimization and Parallelization of Fuzzy Clustering Algorithm Based on the Improved Kmeans++ Clustering

Wenjuan Cheng and Yu Ma

November 11, 2019

Optimization and Parallelization of Fuzzy Clustering Algorithm Based on the Improved Kmeans++ Clustering

Wenjuan Cheng
School of Computer Science and Information Engineering
HeFei University of Technology
HeFei, China

Yu Ma
School of Computer Science and Information Engineering
HeFei University of Technology
HeFei, China

***Abstract*- Fuzzy clustering algorithm is one of the most widely used clustering algorithm in the field of big data. Although fuzzy c-means (FCM) algorithm performs well, it still has some problems like sensitive to initial clustering center and difficult to determine the number of clusters. To solve these problems, we put forward an improved fuzzy clustering algorithm based on kmeans++ algorithm. The improved algorithm optimized the kmeans++ algorithm with the Canopy algorithm, integrated the L2 norm, and parallelized based on spark. Experimental result shows that the improved algorithm performs better on clustering accuracy and computational performance.**

Keywords- Fuzzy C-means; kmeans++; Canopy; Spark; parallelization

Ⅰ. INTRODUCTION

Fuzzy clustering algorithm is an important soft clustering algorithm. Compared with the hard clustering algorithm which stick to "one sample belongs to one cluster", fuzzy clustering algorithm based on membership degree performs better on data description. Among fuzzy clustering algorithms, the fuzzy c-means clustering (FCM) algorithm [1] which improved by the hard c-means clustering (HCM) algorithm is one of the most completely and widely used algorithm. However, the FCM algorithm still has the defects that the number of clusters is difficult to determine and the local extremum is easy to fall into.

To solve these problems, researchers at home and abroad did a lot of research. Han Zhe [2] proposed using genetic algorithm to initialize the cluster center and applied it to image segmentation. Liang Bing [3] proposed using the artificial bee colony algorithm to initialize the cluster center. Some researchers [4-6] proposed using biological evolutionary algorithms to improve fuzzy clustering algorithms, but such algorithms have a large overhead computation. The Canopy algorithm [7] has made good progress in rapidly clustering large-scale and multi-feature data sets, and it is the mainstream method for quickly determining the number of clusters currently. The kmeans++ algorithm [8] which selects cluster center based on Euclidean distance performs well in avoiding local extremum. Bahmani [9] implemented the kmean‖ algorithm for distributed computing based on the kmeans++ algorithm.

This article proposed an improved fuzzy clustering algorithm. First, the algorithm used the L2 norm instead of the Euclidean distance to optimize the distance calculation. Second, the algorithm combined the Canopy algorithm and the kmeans++ algorithm to initialize clustering center. Third, the algorithm was implemented based on the big data platform Spark [10]. Experimental result shows that the improved algorithm performs better on both algorithm accuracy and big data adaptability.

Ⅱ. PRELIMINARY

A. FCM Algorithm

The optimization objective function of the FCM

Foundation: This work was supported by the National Natural Science Foundation of China (Project No. 51274078) and the Quality Engineering of AnHui Province (Project No.2016ckjh141）

algorithm is:

$$J_{FCM}^{m}(U,V,X)=\sum_{i=1}^{c}\sum_{j=1}^{n}u_{ij}^{m}d_{ij}^{2}=\sum_{i=1}^{c}\sum_{j=1}^{n}u_{ij}^{m}\|x_j-v_i\|^2 \quad (1)$$

Where, $U=[u_{ij}]_{c*n}$ is the membership matrix, the value range of u_{ij} is [0,1] , X is the dataset, V is the cluster center matrix, d_{ij}^2 is Euclidean distance, c is the number of clusters, n is the total number of samples, m is the fuzzifification parameter.

In the HCM algorithm, the value of u_{ij} is 0 or 1, which means a sample only belongs to one specific cluster. However, many samples do not have strict classification boundaries because of the ambiguity and uncertainty. The FCM algorithm extends the range of membership degrees (u_{ij} values 0 to 1) to have a better data representation. u_{ij} also should meet the normalization constraints:

$$\begin{aligned}&\sum_{i=1}^{c}u_{ij}=1,(1\le j\le n)\\&u_{ij}\ge 0,(1\le i\le c,1\le j\le n)\end{aligned} \quad (2)$$

The FCM algorithm is an algorithm that iteratively solves the minimum value of the objective function. The iterative formula of the v_i and u_i can be obtained by Lagrangian multiplication:

$$v_i=\frac{\sum_{j=1}^{n}u_{ij}^{m}x_j}{\sum_{j=1}^{n}u_{ij}^{m}} \quad (3)$$

$$u_{ij}=\frac{1}{\sum_{k=1}^{c}(\frac{d_{ij}}{d_{kj}})^{\frac{2}{m-1}}} \quad (4)$$

The execution steps of the FCM algorithm are:

Step1: Initialize the cluster center V. Specify the number of clusters c and the iteration stop threshold ε.

Step2: According to Equation (4), calculate the membership matrix U .

Step3: Calculate and update the cluster center V according to Equation (3).

Step4: Calculate the objective function J_{FCM}^m according to Equation (1). The iteration will stop when the value is less than the threshold. Then we get the membership matrix U and the cluster center matrix V. Otherwise the algorithm will go back to Step2.

B. Improved Canopy-kmeans++ algorithm

FCM algorithm is sensitive to initial clustering center. Most scholars have proposed to use biological evolutionary algorithms (genetic, ant colony, bee colony, bat, etc) to initialize the cluster center. However, such algorithms have large overhead computation and perform poorly on dealing with large data sets. In this article, the Canopy algorithm was used to improve the kmeans++ algorithm to initialize the cluster center, and the L2 norm was used to instead of the Euclidean distance to simplify the distance iterative calculation.

Firstly, to quickly obtaine the number of clusters, we used the Canopy algorithm to cluster the dataset "Roughly". Then we used the kmeans++ algorithm to cluster dataset "Finely" [11]. The L2 norm calculation formula is shown in Equation (5), the Euclidean distance calculation formula is shown in Equation (6). The L2 norm is smaller than the Euclidean distance, and is easier to calculate, which optimizes the distance calculation.

$$\begin{aligned}&d_{L2}=(\sqrt{a_1^2+b_1^2}-\sqrt{a_2^2+b_2^2})^2=\\&a_1^2+b_1^2+a_2^2+b_2^2-2\sqrt{(a_1^2+b_1^2)(a_2^2+b_2^2)}\end{aligned} \quad (5)$$

$$\begin{aligned}&d_{EU}=(a_1-a_2)^2+(b_1-b_2)^2=\\&a_1^2+b_1^2+a_2^2+b_2^2-2(a_1a_2+b_1b_2)\end{aligned} \quad (6)$$

Among them, the center point is (a1, b1) and the sample point is (a2, b2).

The main steps of the cluster center initialization improvement algorithm are as follows:

Step1: Read the data set, convert it to vector and store it in the list, define the distance thresholds T_1 and T_2 ($T_1>T_2$). T_1 and T_2 can be determined by cross-check.

Step2: Randomly selecte a sample point P in the list, and calculate the L2 norm distance between the sample point and the center point of all Canopy subsets, the calculation formula is shown in Equation (7). If the distance is less than T_1, point P will be added to this Canopy, else if P is less than T_2 with any Canopy center, P will be removed from the list.

$$D_{L2}(x,V)=d_{L2}(x,v) \quad (7)$$

Step3: Iterate Step2 until the list is empty, finally the

data set will be divided into class c.

Step4: Calculate the cost of the sample set relative to the cluster center matrix, and calculate the formula as follows:

$$\varphi_X(V) = \sum_{x \in X} {D_{L2}}^2(x, V) \quad (8)$$

Step5: Calculate the probability that the sample is selected as the cluster center. Select it until the quantity is equal to c. The probability calculation formula is as follows:

$$P_x = \frac{{D_{L2}}^2(x, V)}{\varphi_X(V)} \quad (10)$$

Step6: Summarize the cluster center points, and obtain the initialized cluster center matrix.

C. Spark MLlib

To improve the big-data adaptability of the algorithm, the algorithm was parallelized on the distributed computing platform Spark. At present, Apache Spark and Apache Hadoop [12] are the mainstream distributed computing platforms. The reasons for using Spark are as follows:

(1) Spark has a higher computing speed. It saves the intermediate results in the memory, and can better adapt to the machine learning algorithm based on iterative operation.

(2) The core of Spark computing is RDD. Spark provides a rich set of operations for RDD, such as conversion operations, motion operations, and persistence operations.

(3) Currently, Spark has the best open source distributed machine learning library, MLlib.

Since the FCM algorithm has attributed the clustering problem to the mathematical problem (iteratively solving the minimum value of the objective function), there are a large number of vector and matrix operations in the algorithm. Spark MLlib implements the common calculation operations of RDD vectors and matrices by encapsulating Breeze library and BLAS library, including the following contents:

(1) Vector operation.

axpy(a, $\boldsymbol{x}$, $\boldsymbol{y}$) // $\boldsymbol{y} = \boldsymbol{y} + a\boldsymbol{x}$

scal(a, $\boldsymbol{x}$) //multiplication

Vectors.norm($\boldsymbol{x}$, 2) //L2 norm of the vector

Vectors.sqdist($\boldsymbol{x}$, $\boldsymbol{y}$) //squared distance between vectors

(2) Matrix operation.

//Transpose operation, A is matrix

$\boldsymbol{A}$.transpose

//Addition operation, $\boldsymbol{C} = a\boldsymbol{A} + b\boldsymbol{B}$.

gemm(a, $\boldsymbol{A}$, $\boldsymbol{B}$, b, $\boldsymbol{C}$)

//multiplication operation

$\boldsymbol{A}$.multiply($\boldsymbol{B}$)

(3) Distributed matrix operations.

MLlib also provides common matrix operations such as add,multiply and transpose. In addition, literature [13] has also conducted sufficient research on matrix calculations.

III. IMPROVED FCM BASED ON SPARK

A. Algorithm implementation points

(1) Initial clustering center. Initialized the cluster center with the improved kmeans++ algorithm based on Canopy algorithm. Used the L2 norm instead of the Euclidean distance to simplify the distance iterative calculation.

(2) Cache and persist data. FCM is an iterative algorithm. For the data that needs to be read repeatedly, it can be cached or persisted into memory to improve the efficiency of algorithm execution.

(3) Data format conversion. Since the L2 norm was used instead of the Euclidean distance, the sample vector and the L2 norm can be encapsulated to form a new data format for the convenience of calculation. The RDD format conversion is shown in Figure 1. The process of converting sample data into RDD[VectorWithNorm] format is: RDD[String] -> RDD[Vector] -> RDD[Double] -> RDD[Vector, Double)] -> RDD[VectorWithNorm].

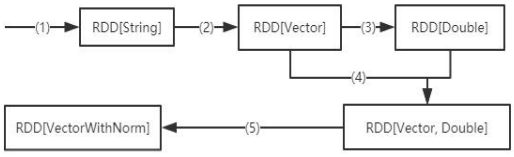

Figure 1. RDD format conversion

(4) Calculate and update the cluster center. When

adding the value of current sample point to the sum of the clusters, the vector addition method “axpy” can be used. When updating the cluster center point, vector multiplication method “scal” can be used.

(5) Accumulator and broadcast variables. During Hadoop's MapReduce, calculate the sum is a common operation. Spark provides method “accumulator” to achieve the summation and the counting of RDD. Spark also provides method “broadcast” to broadcaste variables. During the FCM algorithm, broadcast the cluster center point can save the time of nodes communication and improve the efficiency of the algorithm.

B. Parallelization based on spark

The algorithm based on Spark can improve the robustness, convergence speed and accuracy, especially when dealing with large-scale sample data [14]. The improved fuzzy clustering algorithm was parallelized on Spark.

The execution steps of the improved algorithm based on Spark are as follows:

Step1: Build a Spark object, SparkContext is the entrance of the Spark program.

Step2: Read the sample data from the distributed file storage system HDFS, cache it to the memory. Calculate the L2 norm value of the sample vector and convert it to the RDD[VectorWithNorm] format.

Step3: Initialize cluster center by the improved Canopy-kmeans++ algorithm based on L2 norm. Broadcast the cluster center to each partition.

Step4: Calculate the membership matrix, and use the “mapPartitions” method of Spark RDD to apply the function to each partition.

Step5: Calculate the cluster center, use the method “reduceByKey” and “collectAsMap” of Spark RDD to calculate the sum and count of the sample under the same cluster.

Step6: Calculate the objective function value, if the value is less than the threshold, the iteration stops. Otherwise update the cluster center and return to Step4.

Step7: Get the FuzzyCMeansModel model after iteration.

The algorithm flow chart is shown in Figure 2.

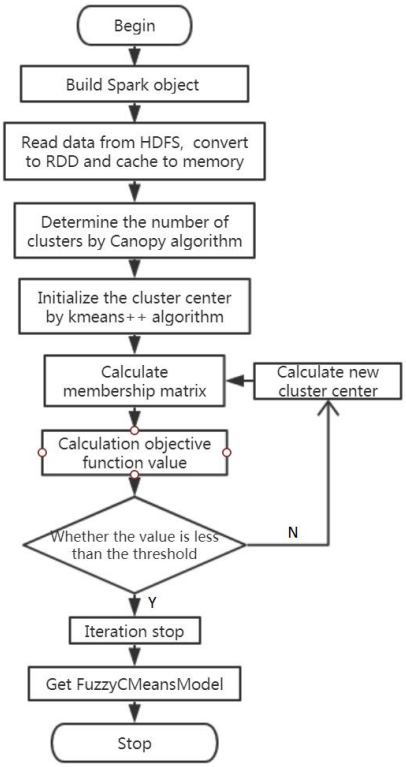

Figure 2. Spark FCM algorithm flow

The pseudo code is shown in algorithm 1.

Algorithm1 Improved algorithm pseudo code

Input：txt //local files

Output：FuzzyCMeansModel //Trained model

1) Read sample data from HDFS

data <- sc.textFile().map()

2) Use the improved Canopy-kmeans algorithm based on L2 norm to initialize the cluster center

centers <- data.Ck()

sc.broadcast(centers) // broadcast center

3) Iterative calculation, calculate the membership and assign sample to the cluster center

for(*i* <- 0 to maxIterations){

data.mapPartitions()

sum <- reduceByKey()

count <- collectAsMap()

newCenter <- scal(1.0 / *fuzzyCount*, *sum*)

if (J_i - J_{i+1} <ε) {break}

else {centers <- newCenter}

}

4) Finally get the FuzzyCMeansModel model

FuzzyCMeansModel <- Spark FCM

Ⅳ. EXPERIMENTAL EVALUATION

A. Experimental Environment and dataset

The Spark cluster runs in a model of "Spark on Yarn". It was built on Hadoop Distributed File System (HDFS) and Resource Scheduling System (Yarn). The Spark cluster consists of six virtual machines created by three PCs. Each PC creates two virtual machines. One virtual machine was used as the master node and the others were used as work nodes. The configuration of each PC is as follows: Win7 system, VM ware 14.0.0 virtual machine software, Intel Core i5-8400 @2.80 GHz CPU, GEFORCE GTX 1050 Ti NIC, 16 GB memory and 2 TB hard drive. The configuration of each virtual machine is as follows: the system version is CentOS 7.4, the Java environment is jdk_1.8.0, the Hadoop version is 2.6.0, the Scala version is 2.10.6, and the Spark version is 1.6.0.

The experimental data set was taken from the UCI machine learning library, including iris (Iris flower dataset), Wine (wine dataset) and KDDcup (simulated network attack dataset). The specific features are shown in Table 1.

Table 1. Data set from UCI

Data set	Number of samples	Number of attributes	Number of clusters
iris	150	4	3
Wine	178	13	3
KDDcup	4×10^4	42	23

B. Comparison of clustering effects

For the convenience of recording, the improved fuzzy clustering algorithm proposed in the paper will be abbreviated as FCM-Ck. Analyze the clustering effect of improved algorithm by testing cluster center results and clustering accuracy. The algorithms for comparison experiments are as follows: FCM algorithm, IABC-KGFCM artificial algorithm based on bee colony (literature [3] algorithm), kmeans++ algorithm. Cluster center results experiment using iris as test data set. The actual clustering center of iris data set is: {(5.00,3.42,1.46,0.24), (5.93,2.77,4.26,1.32), (6.58,2.97,5.55,2.02) }, The clustering center results of the four algorithms under the iris data set are shown in Table 2.

Table 2. Comparison of cluster center results

Algorithm	Cluster center results	Distance from the actual cluster center
FCM	v_1 =(5.010,3.411,1.469,0.254)	0.021
	v_2=(5.891,2.762,4.379,1.411)	0.155
	v_3=(6.810,3.049,5.671,2.049)	0.273
IABC-KGFCM	v_1 =(5.008,3.415,1.450,0.253)	0.017
	v_2=(5.972,2.786,4.133,1.394)	0.152
	v_3=(6.461,2.779,5.673,2.041)	0.257
kmeans++	v_1 =(5.006,3.423,1.467,0.225)	0.018
	v_2=(5.896,2.779,4.411,1.311)	0.155
	v_3=(6.741,3.102,5.687,2.001)	0.250
FCM-Ck	v_1 =(5.003,3.419,1.471,0.231)	0.014
	v_2=(5.890,2.778,4.326,1.396)	0.108
	v_3=(6.711,2.895,5.694,2.018)	0.209

From Table 2, the FCM-Ck algorithm has a higher clustering accuracy. It is closer to the actual cluster center than FCM algorithm, IABC-KGFCM algorithm and kmeans++ algorithm. That is due to the improvement of the cluster center initialization method. The sample point farther from the center are more easily selected as the cluster center. FCM-Ck algorithm performs better in mitigates the problem of falling into the local optimal solution.

The cluster accuracy experiment is based on the iris, Wine, and KDDcup data sets. Divide the data set into a training set and a test set in a scale of 0.7:0.3. The accuracy of the algorithm is calculated as $P=(N-W)/N$, Where N is the total number of samples, and W is the number of error clustering samples.The algorithm accuracy rate results are shown in Figure 3.

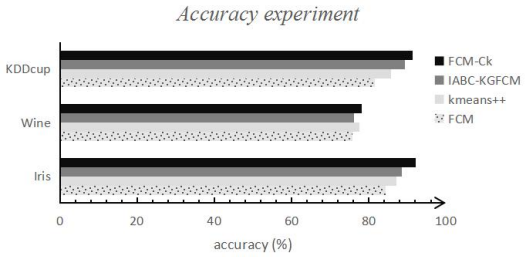

Figure 3. experiment on algorithm accuracy

According to Figure 3, the accuracy of

kmeans++ algorithm is higher than FCM, which is due to the method of selecting the cluster center based on the Euclidean distance. FCM-Ck algorithm performs better than kmeans++ algorithm and IABC-KGFCM algorithm, moreover, compared to small-scale data sets (such as iris and Wine), the improvement of accuracy is more obvious in the large-scale data set (like KDDcup).

C. Algorithm efficiency analysis

Analyze the running time of each algorithm on different data sets. The results are shown in Table 3.

Table 3. Average clustering time

Algorithm	Data set		
	iris	Wine	KDDcup
FCM	0.72	0.74	61.72
kmeans++	0.67	0.71	56.64
IABC-KGFCM	0.81	0.84	72.37
FCM-Ck	0.64	0.67	29.47

According to Table 3 and Figure 3, compared with FCM, the IABC-KGFCM algorithm based on artificial bee colony has improved the accuracy. But it also paid a certain cost of time, and it did not perform well when dealing with large-scale data set. The FCM-Ck algorithm has significantly improved the time performance, which is due to its distance calculation strategy and Spark-based parallel computing strategy.

On the data set KDDcup, The average number of iterations for each algorithm to converge is shown in Table 4. Among them, the FCM-Ck algorithm and the kmeans++ algorithm have the least number of iterations. That indicates their initial cluster center method is more reasonable and more accessible to the actual cluster center.

Table 4. Number of iterations

Algorithm	Data set		
	iris	Wine	KDDcup
FCM	27	31	64
kmeans++	9	8	16
IABC-KGFCM	11	14	22
FCM-Ck	6	6	13

Speed ratio and expansion ratio are also important indicators to measure the efficiency of parallel algorithms [15]. Speed ratio is the ratio of the time spend on single node and multiple nodes. It is used to measure the time performance improvement of the algorithm in the cluster. The calculation formula for speed ratio is:

$$S = \frac{T_1}{T_p} \qquad (11)$$

Where, S is the speed ratio, P is the number of cluster nodes, T_1 is the running time of a single node, T_p is the running time of p nodes.

Expansion ratio is an important criterion for measuring the scalability of parallel algorithms. The higher the expansion ratio, the better the scalability of parallel algorithms. The formula for calculating the expansion ratio is:

$$E = \frac{S}{P} \qquad (12)$$

The experimental data set of the speed ratio and the expansion ratio is taken from the UCI machine learning library, and the test results are shown in Figure 4 and Figure 5.

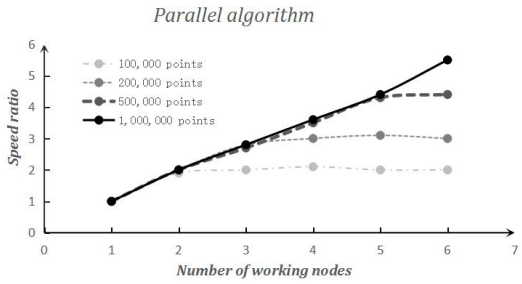

Fig.4 speed ratio experiment

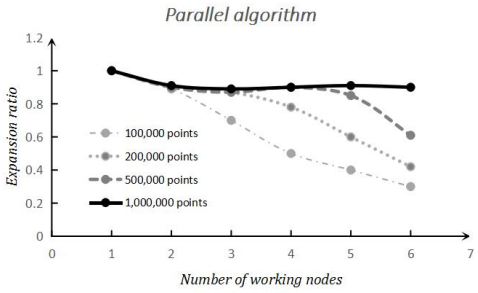

Fig.5 extension ratio experiment

According to Figure 4 and Figure 5, when processing small-scale data (below 500,000), the speed ratio increases linearly and then stabilizes. Because fewer nodes are sufficient to process the data

of this scale, and the acceleration of adding nodes is no longer obvious. Therefore, the expansion ratio will also decrease when the nodes increases to a certain number. However, when dealing with big data (up to 1 million), FCM-Ck algorithm improves the performance of the algorithm obviously. The speed ratio increases linearly with the increase of the nodes' number, the expansion ratio tends to 0.9 and basically stable. That is, the larger size of the data, the higher performance of the improved algorithm.

V. CONCLUSION

Aiming at the problems that the fuzzy c-means algorithm easy to fall into local optimum and difficult to determine the clustering numbers, this article proposed an improved fuzzy clustering algorithm based on kmeans++. The improved algorithm used Canopy algorithm to quickly determine the number of clusters, used kmeans++ algorithm to enhance the global search ability of the algorithm, introduced L2 norm instead of Euclidean distance to simplify distance calculation, and was parallelized based on big data platform Spark. The improved algorithm was compared with FCM, IABC-KGFCM and kmeans++ algorithm through experiments. Experimental results show that the improved algorithm performs better on algorithm accuracy, computational performance and big data adaptability.

REFERENCES

[1] BEZDEK J C. A Convergence theorem for the fuzzy ISODATA clustering algorithms[J]. IEEE Trans Pattern Anal Mach Intell, 1980,2(1):1-8.

[2] HAN Zhe, LI Deng-ao, ZHAO Ju-min, CHAI Jing.Image segmentation algorithm based on improved genetic fuzzy clustering and level set[J].Computer Engineering and Design,2019,40(05):1390-1393+1412.

[3] LIANG Bing, XU Hua. Kernel fuzzy C-means clustering based on improved artificial bee colony algorithm[J]. journal of Computer Applications, 2017, 37(09): 2600-2604.

[4] TAN Ling-long, WANG Qing, LI Chang-kai.Optimization of Fuzzy C-Means Clustering Based on Adaptive Ant Colony Algorithm[J].Measurement & Control Technology, 2018,37(05):46-50.

[5] CUI Fangyi, JING Xiaoyuan, DONG Xiwei, WU Fei, SUN Ying.Fuzzy Clustering Based on Adaptive Bat Algorithm Optimization and Its Application[J].Computer Engineering and Applications,2019,55(07):16-22.

[6] Zhong Weiwei, Liu Liping, Wang Fangzheng. Research on Cloud Security Algorithm Based on Immunology and Fuzzy Clustering[J]. Network Security Technology & Application,2019(10):41-44.

[7] Mccallum A, Nigam K, Ungar L H.Efficient clustering of high-dimensional data sets with application to reference matching[A].ACM SIGKDD[C].2000:169~178.

[8] Arthur D, Vassilvitskii S.k-means++:The advantages of careful seeding[C]//Proceedings of the Eighteenth Acm-Siam Symposium on Discrete Algorithms.New Orleans, Louisiana, USA.2007:1027-1035.

[9] BAHMANI B, MOSELEY B, VATTANI A. Scalable k-means++[J]. Proceedings of the VLDB Endowment, 2017,5(7):622-623.

[10] ZAHARIA M, CHOWDHURY M, DAS T, et al. Resilient distributed datasets:a fault-tolerant abstraction for in-memory cluster computing[C]//Proceedings of the9th USENIX Conference on Networked Systems Design and Implementation. Berkeley, CA:USENIX Association, 2012:141-146.

[11] Yu Changjun. Research and implementation on fuzzy C-means algorithm[D]. Wuhan University of Technology, 2014.

[12] White T. Hadoop: the definitive guide[M]. Beijing: Tsinghua University Press, 2015.

[13] WANG Guilan, ZHOU Guoliang, SA Churila, ZHU Yongli. Parallel fuzzy C-means clustering algorithm in Spark[J]. journal of Computer Applications,2016,36(02): 342-347.

[14] WU Yun-long, LI Ling-juan. Implementation and Application of Fuzzy Clustering Algorithm Based on Spark[J]. Computer Technology and Development,2019, 29(01):130-134.

[15] Liu De-chao. Research on parallelization of clustering algorithm based on MapReduce[D]. North China Electric Power University,2016.