Fast K-Means Clustering for Very Large Datasets Based on MapReduce Combined with a New Cutting Method

Duong Van Hieu and Phayung Meesad

Abstract. Clustering very large datasets is a challenging problem for data mining and processing. MapReduce is considered as a powerful programming framework which significantly reduces executing time by dividing a job into several tasks and executes them in a distributed environment. K-Means which is one of the most used clustering methods and K-Means based on MapReduce is considered as an advanced solution for very large dataset clustering. However, the executing time is still an obstacle due to the increasing number of iterations when there is an increase of dataset size and number of clusters. This paper presents a new approach for reducing the number of iterations of K-Means algorithm which can be applied to very large dataset clustering. This new method can reduce up to 30 percent of iterations while maintaining up to 98 percent accuracy when tested with several very large datasets with real data type attributes. Based on the significant results from the experiments, this paper proposes a new fast K-Means clustering method for very large datasets based on MapReduce combined with a new cutting method (abbreviated to FMR.K-Means).

Keywords: MapReduce, K-Means, Fast K-Means for very large datasets.

1 Introduction

In the big data era, collected datasets become very large and complex which not only bring potentially and highly undiscovered values to business, management, scientific researches but also lead to a challenging problem for mining and processing tasks [1]. One of the most concerned issues of partitioning clustering methods is

Duong Van Hieu · Phayung Meesad
Faculty of Information Technology
King Mongkut's University of Technology North Bangkok
Bangkok 10800, Thailand
e-mail: duongvanhieu@tgu.edu.vn, pym@kmutnb.ac.th

coping with very large datasets. Various methods were proposed to use for dealing with very large dataset clustering such as dataset size reduction, using representative samples, parallelization, and better initial center selection [2, 3, 4]. However, these methods can not completely solve the problem "process masses of heterogeneous data within a limited time" [5]. Among the data clustering techniques, which are statistical classification techniques to discover the nature of set groupings of data which fall into different groups, K-Means has gained great interest from researchers because of its advantages in data analyzing but has limitations when working with very large datasets nowadays [6]

The K-Means algorithm has strength and performance in terms of working capability to deal with large datasets by utilizing advanced resources and network infrastructure including multicore machines, computer clusters. Parallel K-Means utilizes the available resources of multicore machines [7], and parallel K-Means based on MapReduce [8] is considered as an advanced model for boosting large dataset processing capability on distributed environments, and gained some successes in large dataset clustering [9, 10, 11]. Although parallel K-Means based on MapReduce has made a new trend referring to large dataset clustering, it is believed that this model needs to be improved to defeat the challenging problems of data intensive applications [1].

To reduce executing time of the K-Means algorithms when working with large datasets, Christopher [12] proposed to apply two termination conditions separately or combine each with a fixed number of iterations, α, to ensure that the algorithm will be terminated in an acceptable period of time. These terminative criteria are when RSS falls below a threshold σ and when decrease in RSS falls below a threshold θ. However, which values the α, σ and θ should take is still a question. On the other hand, some sampling methods have been proposed including clustering using representative (CURE) [13], coresets for K-Means and K-Median [14]. These sampling methods are efficient in terms of reducing executing time. However, it is believed that the accuracy is still not significantly high due to working on proportions instead of the whole collected datasets.

Based on the problems mentioned above, this paper proposes a new approach for reducing executing time of the K-Means clustering by cutting off a number of iterations in clustering very large datasets. To be more adequate to the increasing in size of datasets, this paper also proposed a fast K-Means clustering algorithm for very large dataset clustering based on the MapReduce combined with a new iteration cutting method, called FMR.K-Means.

The contents of this paper are organized into five sections. Related work is presented in Section 2. Section 3 explains the proposed approach for cutting off iterations of the K-Means algorithm for very large dataset clustering. Experimental analysis is shown in Section 4. Finally, conclusion and future work will are covered in the last section.

2 Related Work

This section presents the related literature review which will be applied in this study including the K-Means algorithm, parallel K-Means based on the MapReduce programming model, and cluster validity based on external information.

2.1 K-Means Algorithm

The K-Means clustering is one of the most well-known clustering techniques in the machine learning discipline. This method tries to cluster a provided dataset X having N samples into K clusters in such a way that intra-cluster similarity is high whereas inter-cluster similarity is low. Firstly, K samples from the provided dataset will be selected to be initial cluster centers. Secondly, each remaining sample is assigned to the closest cluster based on the distance between the sample and the cluster center. Thirdly, the new center for each cluster is then calculated using information of samples, which belong to that cluster. This process iterates assigning samples to clusters and recalculating center of clusters until the terminate criterion is satisfied [12, 15].

The K-Means clustering is quite easy to deploy and works well with numerical data types. However, it is inadequate to very large datasets due to intensive calculation. It needs NxKxI distance calculations to calculate distances between data samples to cluster centers where N is number of samples in the provided dataset, K is number of clusters, and I is number of iterations. There will be a significantly high increase in executing time when there is an increase in dataset size or number of clusters, especially when N increases highly.

2.2 Parallel K-Means Based on MapReduce

Parallel K-Means based on the MapReduce was proposed in [9, 16] including three functions that are (1) a map function which takes care of calculating distance from each data sample to clusters and assigns this data sample to the closest cluster, (2) a combine function which calculates local centers before sending them to the reducing function, and, (3) a reduce function which obtains local centers and calculates global centers of each cluster.

Theoretically, regardless of data transferring time, the executing time will be reduced W times if this model runs on a machine with W cores or a cluster having W workers. However, this model can only solve the challenge in terms of "divide and conquer". The challenging problem is still there when the data size increases exponentially and the pressure from processing very large dataset within a limited time increases [5].

2.3 Cluster Validity

Evaluation of clustering results is one of the most important issues in cluster analysis. Cluster validity is a technical term for a procedure of evaluating the results of a clustering algorithm. Three most used approaches to investigate cluster validity are external criteria based, internal criteria based, and relative criteria based. This section focuses on the first approach because it is the best choice to evaluate results of the FMR.K-Means algorithm. Cluster validity based on external criteria which evaluates the results of a clustering algorithm based on a pre-specified structure [17, 18]. The common use of external criteria is to compare clustering structure C to an independent partition of data P. Let

- $C=\{C_1, C_2,...,C_k\}$be a clustering structure of a provided dataset X.
- $P=\{P_1, P_2, ...,P_s\}$be a predefined partition of X.
- SS be points belong to the same cluster of C and to the same group of P.
- SD be points belong to the same cluster of C and to different group of P.
- DS be points belong to different cluster of C and to the same group of P.
- DD be points belong to different cluster of C and to different group of P.
- a, b, c, d is value of SS, SD, DS, DD, respectively.
- M be the maximum number of pairs in the dataset.

The degree of similarity between C and P can be measured by:

$$M = a + b + c + d. \tag{1}$$

1. Rand Statistic

$$R = (a + d)/M. \tag{2}$$

2. Jaccard Coefficient

$$J = a/(a + b + c). \tag{3}$$

Value of R, J can be either 0 or 1 corresponding to completely different or similar. The larger value R, J gets, the higher similar between C and P.

3 Fast K-Means Clustering for Very Large Datasets Based on MapReduce Combined with a New Cutting Method

This section firstly states the obstacles of very large dataset clustering tasks in terms of executing time. Based on these barriers and preliminary experiment results, a new approach for cutting off last iterations will be proposed to use for high dimensional very large datasets with real data type attributes. Lastly, a fast K-Means algorithm based on the MapReduce combined with a new cutting method is proposed to use for clustering very large datasets.

3.1 The Obstacles of Very Large Datasets Clustering Using K-Means

It is clearly that the K-Means clustering algorithm is a well-known clustering method but two obstacles exist for very large datasets clustering. The first obstacle is computational complexity of distance calculations, which calculates distances between data samples to clusters. This difficulty can be overcome by applying the

MapReduce model to distribute computations to multiple workers in a distributed environment. However, this obstacle is still there when data size increases exponentially.

The second obstacle is the number of iterations which significantly increases when the number of sample data increases. This problem may be solved by using two-stages K-Means algorithm or K-Means++ algorithm [19]. K-Means++ consists of two steps: the first step is to select better initial centroids and the second step is the K-Means. This K-Means variant may reduce the number of iterations of K-Means by applying a new technique for selecting the better initial centroids in the initial stage. However, it also needs more time for this first step due to high computational complexity. In case the best initial centers selected, it also needs more time to execute a large number of iterations at the second stage.

To solve that problem, a new cutting method will be proposed to cut off last iterations of the K-Means clustering, which is presented in the next section.

3.2 A New Proposed Method for Reducing a Number of Iterations

This research deals with a problem of how to cut off a number of iterations of the K-Means clustering in order to reduce executing time while maintaining significantly high accuracy. The preliminary experiments show that the last iterations have the least contribution to the percent of correctly clustered objects and the number of iterations will significantly increases if there is a big increase in data size [12].

This section aims to propose a new method called cutting off the last iterations based on differences between centers of each cluster of two adjacent iterations. This proposed method can be explained as follows.

- Let
  - x_i=(x_{i1}, x_{i2}, ..., x_{iP}) $\in R^p$ be a sample having P attributes.
  - X={x_1, x_2, .., x_N} be a provided dataset having N samples, which need to be clustered into K groups.
  - $c_i^{(j-1)}$, $c_i^{(j)}$ $\in R^p$ be center of cluster i at the (j-1)th and the jth iteration.
  - v = (v_1, v_2, ..., v_P) $\in R^p$ be a vector can be driven from X.
- Define a new measure called vector Norm of X
  - $Norm(X)$=(v_1, v_2, ..., v_P)

where

$$v_j = max(x_{ij}) - min(x_{ij})\, with\, j = 1,2,..P; i = 1,2,..N\,. \tag{4}$$

- Define a new terminative condition
  - $\Delta \leq \varepsilon$.

where

$$\Delta = \frac{|c_i^{(j)} - c_i^{(j-1)}|}{Norm(X)}. \tag{5}$$

$$\varepsilon = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_P) \in R^P \, with \, \varepsilon_i = 0.01x2^{(-m)}. \quad (6)$$

- m is a fault tolerance degree, it can be from 0 to positive infinitive. The higher the tolerance degree is used, the higher accuracy is obtained.

The K-means algorithm with the new cutting off last iteration method is depicted as Algorithm 1.

Algorithm 1. K-Means (X, K, m)

```
Input:  A given dataset $X$, number of clusters $K$, fault tolerance degree $m$
Output: A set of centers, index of clusters that each object belongs to
  1.  Calculate fault tolerance vector $\varepsilon$
  2.  Calculate $Norm\ (X)$
  3.  Initialize $K$ centers
  4.  Assign each object to the closest cluster
  5.  Calculate new centers
  6.  Calculate vector $\Delta = \frac{|c_i^{(j)} - c_i^{(j-1)}|}{Norm(X)}$
  7.  Check stop condition $\Delta \leq \varepsilon$
  -   If the stop condition is not satisfied, update centers and repeat from step 4
  -   Otherwise, return a set of centers, index and stop
```

3.3 A Fast K-Means Algorithm Based on MapReduce Combined with a New Cutting Method (FMR.K-Means)

The proposed method for cutting off a number of last iterations will be integrated into the Parallel K-Means algorithm based on the MapReduce framework which was proposed in [9] to tackle very large datasets, depicted as Algorithm 5. Algorithm 2 is used to calculate distances from each object to all centers and assigns that object to the closest cluster. Algorithm 3 is devoted to local centroid calculations. It obtains inputs from map function instances and calculates local centers of clusters associated with the number of objects assigned to each cluster. Algorithm 4 is devoted to global centroid calculations. It obtains inputs from combine functions and produces global centroids. Algorithm 5 is a fast K-means based on MapReduce combined with the proposed method for cutting off a number of last iterations (abbreviated to FMR.K-Means).

4 Experimental Analysis

This section describes the experiment framework which is carried out by the proposed algorithm. The experiment results will be evaluated in terms of four criteria including percent of reduced iterations, percent of correctly clustered objects called accuracy, Rand statistic and Jaccard Coefficient measures.

```
Algorithm 2. Map (<key1, value1>, global centers)
Input:  A list of <key1, value1> pairs, a list of K global centers. Where key1 is
        position and value1 is content of object.
Output: A list of <key2, value2>. Where key2 is composition of key1 and index of
        clusters that each object assigned to, key2 is content of object.
  1.  Initialize a list of <key2, value2> pairs.
  2.  Get a pair <key2, value2> that has not been updated, called $x_i$
  3.  Calculate distances between $x_i$ to global centers
  4.  Update $x_i$
  5.  Check stop condition
  -   If exist a pair <key2, value2>has not been updated, repeat from step 2
  -   Otherwise, return a list of <key2, value2>, and stop.
```

```
Algorithm 3. Combine (<key2, value2>)
Input:  A list of <key2, value2>.
Output: A list of <key3, value3>. Where key3 is index of clusters, value3 is local
        center associated with number of objects belong to that cluster.
  1.  Initialize a list of <key3, value3> pairs.
  2.  Get a pair <key3, value3> that has not been updated, called $x_i$
  3.  Calculate value3
  4.  Update $x_i$
  5.  Check stop condition
  -   If exist a pair <key3, value3>has not been updated, repeat from step 2
  -   Otherwise, return a list of <key3, value3>, and stop.
```

```
Algorithm 4. Reduce (<key3, value3>)
Input:  A list of <key3, value3>.
Output: A list of <key4, value4>. Where key4 is index of clusters, value4 is global
        center of clusters.
  1.  Initialize a list of <key4, value4> pairs.
  2.  Get a pair <key4, value4> that has not been updated, called $x_i$
  3.  Calculate value4
  4.  Update $x_i$
  5.  Check stop condition
  -   If exist a pair <key4, value4>has not been updated, repeat from step 2
  -   Otherwise, return a list of <key4, value4>, and stop.
```

Algorithm 5. FMR.K-Means (X, K, m, W)

```
Input:  A provided dataset $X$, number of clusters $K$, fault tolerance degree $m$, number
        of workers $W$
Output: A list of global centers and indexes of clusters that each object assigned to.
 1.  (at master worker)
 -   Calculate fault tolerance vector $\varepsilon$
 -   Calculate vector $Norm(X)$
 -   Initialize $K$ centers
 -   Form a list of <key1, value1> from $X$, called List1
 -   Divide List1 into $W$ block and distribute them to $W$ workers
 2.  (at all workers)
 -   Run map function to produce <key2, value2> pairs
 -   Run combine function to produce <key3, value3> pairs
 3.  (at master worker)
 -   Get results from all workers
 -   Form a list of indexes
 -   Form a list of <key3, value3> pairs
 -   Run reduce function to produce <key4, value4> pairs are global centers
 4.  (at master worker)
 -   Calculate vector $\Delta = \frac{|c_i^{(j)} - c_i^{(j-1)}|}{Norm(X)}$
 -   Update global centers
 -   If $\Delta \leq \varepsilon$ , return a list of indexes, a list of centers, and stop
 -   Otherwise, repeat from step 2.
```

4.1 Experiment Framework

The experimentations were performed on two core and four core machines. All code was written in Matlab. Datasets used in these experiments were datasets extracted from YouTube Multiview Video Games datasets [20] and Daily and Sports Activities datasets [21] obtained from the UCI Machine Learning Repository. The basic information of datasets is illustrated in Table 1 below. Each dataset was tested at least 100 times with 9 values of fault tolerance degree m from 0 to 8 with the FMR.K-Means. Average values will be used to analyze efficiency of the new method.

4.2 Results Analysis

In this section, the term accuracy is defined as the ratio of number of samples that are correctly clustered compared to the results without using fault tolerant variable and total number of samples of the provided dataset. Firstly, experimental results from datasets extracted from the Daily and Sport Activity database, named Dataset1, Dataset2, Dataset3, Dataset4 which are listed in Table 1 show that the average number of iterations increases considerably when there is an increase in the number of data samples, illustrated in Fig 1. The increasing in number of iterations will lead to increasing in executing time when there is an increase in number of samples in

Table 1 Information about experiment datasets

Dataset	Size	No. of Samples	No. of Attributes	No. of Clusters
vision_misc	706 MB	97,935	838	31
vision_hist_motion_estimate	706 MB	97,935	838	31
vision_hog_features	706 MB	97,935	838	31
Dataset1	128 MB	285,000	45	3
Dataset2	257 MB	570,000	45	3
Dataset3	467 MB	1,140,000	45	3
Dataset4	934 MB	2,280,000	45	3

datasets. Moreover, when applying this new proposed cutting method to those four datasets, results also show that the higher fault tolerant degree gained the higher accuracy with lower reduction of iterations which is illustrated in Fig 2.

Secondly, experiment results obtained from the testing of three datasets referring to YouTube Multiview Video Game show when the larger size dataset has the smaller fault tolerant degree this should be selected, as illustrated in Fig 3.

For cluster validity, Rand Statistic and Jaccard Coefficient measures are used to show that the results of the proposed method are highly similar to the results of K-means clustering without using fault tolerant variable. In this case, the predefined partition P=$\{P_1, P_2, \ldots, P_s\}$ is a clustered result without using fault tolerant degree m, or the algorithm will stop iterating when there is no change in centroids. The clustering structure C=$\{C_1, C_2, \ldots, C_k\}$ which is used to compared P is the clustered result with using m. Tables 2 and 3 show percentage of reduced iterations, accuracy; Rand Statistic and Jaccard Coefficient measure values.

Table 2 Results tested with Daily and Sport Activities databases

Dataset	No. of Clusters	Fault tolerance degree (m)	Reduced Iterations	Accuracy	Rand Statistic Measure	Jaccard Coefficient Measure
Dataset1	3	7	42.49%	99.09%	0.9912	0.9909
		8	28.48%	99.77%	0.9977	0.9977
Dataset2	3	7	42.42%	93.44%	0.9452	0.9344
		8	25.59%	99.79%	0.9978	0.9978
Dataset3	3	7	63.29%	95.79%	0.9637	0.9579
		8	47.78%	99.43%	0.9943	0.9942
Dataset4	3	7	67.04%	99.06%	0.9907	0.9906
		8	63.07%	99.43%	0.9944	0.9944

Note: m is fault tolerance degree, R measure is Rand statistic measure, and J measure is Jaccard coefficient measure.

Table 3 Results tested with YouTube Multiview Video Games database

Dataset	No. of Clusters	m	Reduced Iterations	Accuracy	R Measure	J Measure
vision_misc	31	4	12.27%	99.29%	0.9983	0.9983
		5	3.47%	99.93%	0.9998	0.9998
vision_hist_motion_estimate	31	4	22.50%	97.70%	0.9962	0.9953
		5	8.94%	99.83%	0.9996	0.9996
vision_hog_features	31	4	60.94%	88.66%	0.9865	0.9838
		5	41.49%	96.19%	0.9957	0.9954

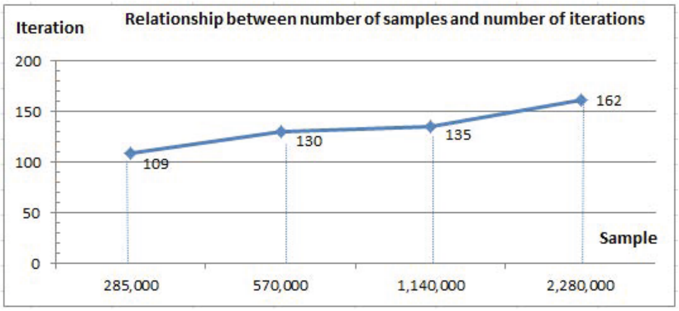

Fig. 1 Relationship between number of samples and number of iterations

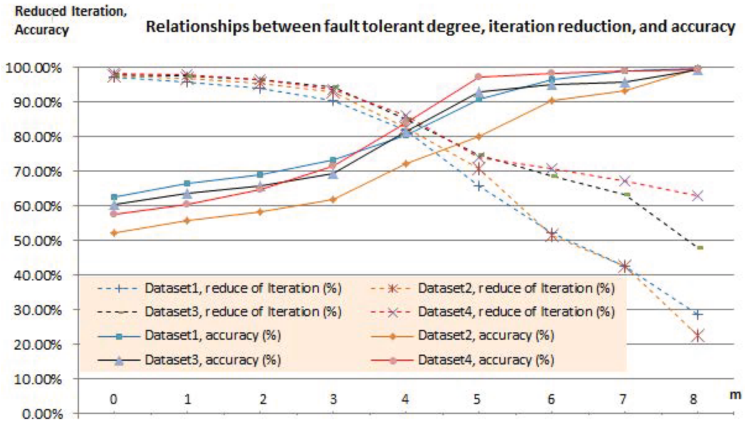

Fig. 2 Relationship between fault tolerant degree, reduced iteration, and accuracy tested with four datasets extracted from Daily and Sport Activities datasets

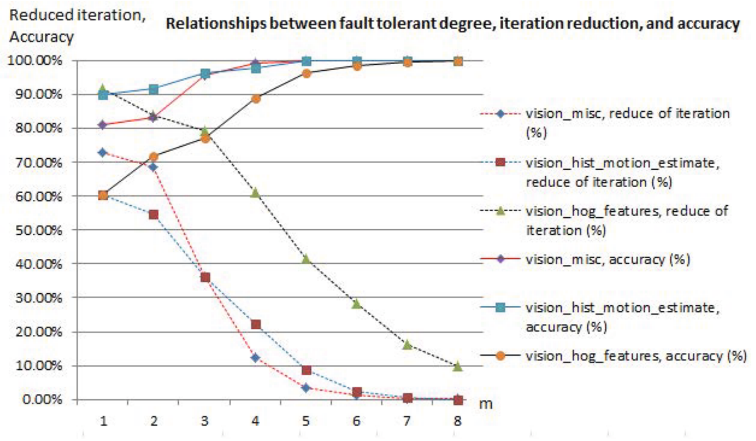

Fig. 3 Relationship between fault tolerant degree, reduced iteration, accuracy tested with three datasets from YouTube Multiview Video Games database

5 Conclusion and Future Work

The experiment results tested with four datasets extracted from Daily and Sport Activities and results tested from three datasets extracted from YouTube Muliview Video Game show that when m=8 and m=4 are selected, respectively, the proposed method can reduce up to 30% of iterations, which is equivalent to 30% of executing time, while maintaining up to 98% accuracy. It can be concluded that this proposed method for cutting off iterations can reduce considerably the number of iterations of K-Means algorithm while maintaining high accuracy. This method of integration with the MapReduce programming model is an appropriate choice for very large dataset clustering jobs. Compared to the previous methods such as clustering using representative (CURE)[8], coresets for K-means and K-median[9], this new method can be considered as more significant because it can work with entire datasets, significantly reduce executing time, and provide high accuracy. For future work, to estimate fault tolerant degrees automatically based on the size of datasets such as number of samples, number of attributes, more experiments need to be carried out with numerous datasets.

References

[1] Philip Chen, C.L., Zhang, C.-Y.: Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. Information Sciences (in press, 2014)
[2] Barioni, M.C.N., Razente, H., Marcelino, A.M.R., Traina, A.J.M., Traina, C.: Open issues for partitioning clustering methods: an overview. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 4, 161–177 (2014)
[3] Hadian, A., Shahrivari, S.: High performance parallel k-means clustering for disk-resident datasets on multi-core CPUs. The Journal of Supercomputing, 1–19 (2014)

[4] Bharill, N., Tiwari, A.: Handling Big Data with Fuzzy Based Classification Approach. In: Jamshidi, M., Kreinovich, V., Kacprzyk, J. (eds.) Advance Trends in Soft Computing. STUDFUZZ, vol. 312, pp. 219–227. Springer, Heidelberg (2014)
[5] Chen, M., Mao, S., Zhang, Y., Leung, V.M.: Chapter 1. Introduction. In: Big Data, pp. 1–10. Springer, Heidelberg (2014)
[6] Jain, A.K.: Data Clustering: 50 Years Beyond K-means. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 3–4. Springer, Heidelberg (2008)
[7] Stoffel, K., Belkoniene, A.: Parallel k/h-Means Clustering for Large Data Sets. In: Amestoy, P.R., Berger, P., Daydé, M., Duff, I.S., Frayssé, V., Giraud, L., Ruiz, D. (eds.) Euro-Par 1999. LNCS, vol. 1685, pp. 1451–1454. Springer, Heidelberg (1999)
[8] Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM 51, 107–113 (2008)
[9] Zhao, W., Ma, H., He, Q.: Parallel K-Means Clustering Based on MapReduce. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) Cloud Computing. LNCS, vol. 5931, pp. 674–679. Springer, Heidelberg (2009)
[10] Lin, C., Yang, Y., Rutayisire, T.: A Parallel Cop-Kmeans Clustering Algorithm Based on MapReduce Framework. In: Wang, Y., Li, T. (eds.) Knowledge Engineering and Management. AISC, vol. 123, pp. 93–102. Springer, Heidelberg (2011)
[11] Lv, Z., Hu, Y., Zhong, H., Wu, J., Li, B., Zhao, H.: Parallel K-means clustering of remote sensing images based on mapReduce. In: Wang, F.L., Gong, Z., Luo, X., Lei, J. (eds.) Web Information Systems and Mining. LNCS, vol. 6318, pp. 162–170. Springer, Heidelberg (2010)
[12] Manning, C.D., Raghavan, P., Schütze, H.: K-Means. In: An Introduction to Information Retrieval. Cambridge University Press (2009)
[13] Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. SIGMOD Rec. 27, 73–84 (1998)
[14] Har-Peled, S., Mazumdar, S.: On coresets for k-means and k-median clustering. Presented at the Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, Chicago, IL, USA (2004)
[15] Jain, A.K., Dubes, R.C.: Chapter 3. Clustering Methods and Algorithms. In: Algorithms for Data Clustering, vol. Computer Science. Prentice Hall (1988)
[16] Anchalia, P.P., Koundinya, A.K., Srinath, N.K.: MapReduce Design of K-Means Clustering Algorithm. In: 2013 International Conference on Information Science and Applications (ICISA), pp. 1–5 (2013)
[17] Dom, B.E.: An Information-Theoretic External Cluster-Validity Measure. In: The Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI 2002), Alberta, Canada, pp. 137–145 (2012)
[18] Wagner, S., Wagner, D.: Comparing Clusterings - An Overview. Institute of Theoretical Informatics (2007)
[19] Xu, Y., Qu, W., Li, Z., Min, G., Li, K., Liu, Z.: Efficient k-means++ Approximation with MapReduce. IEEE Transactions on Parallel and Distributed Systems PP, 1–10 (2014)
[20] UCI. YouTube Multiview Video Games Dataset, `http://archive.ics.uci.edu/ml/datasets/YouTube+Multiview+Video+Games+Dataset`
[21] UCI. Daily and Sports Activities, `http://archive.ics.uci.edu/ml/datasets/Daily+and+Sports+Activities`