A Parallel Clustering Algorithm for Power Big Data Analysis

Xiangjun Meng[1], Liang Chen[2], and Yidong Li[3(✉)]

[1] State Grid Shandong Power Company, Jinan, Shandong, China
[2] Shandong Luneng Software Technology, Jinan, Shandong, China
[3] School of Computer and Information Technology,
Beijing Jiaotong University, Beijing, China
ydli@bjtu.edu.cn

Abstract. With the fast development of information technology, the power data is growing at an exponentially speed. In the face of multi-dimensional and complicated power network data, the performance of the traditional clustering algorithms are not satisfied. How to effectively cope with the power network data is becoming a hot topic. This paper proposes a parallel implement of K-means clustering algorithm based on Hadoop distributed file system and Mapreduce distributed computing framework to deal this problem. The experimental results show that the performance of our proposed algorithm significantly outperforms the traditional clustering algorithm and the parallel clustering algorithm can significantly reduce the time complexity and can be applied in analyzing and mining of the power network data.

Keywords: Parallel algorithm · K-means clustering · Power data

1 Introduction

Clustering [5] is one of the most hot issues in data mining research. It is the process of partitioning data objects into subsets. Each subset is a cluster [11], so that the objects in the cluster are similar to each other, but are not similar to the objects in other clusters. A set of clusters generated by the cluster analysis is called a cluster. With the continuous development of the electric power industry and the popularization of database technology, in the electric power industry, a large amount of data [6, 9] is accumulated in different forms. Then, how to store and utilize these data effectively and how to dig out valuable information from the massive data become problems to be solved. In the face with massive data, the existing data mining algorithms have a lot of problems in time complexity and space complexity. For this problem, the solution is to apply the parallel method to the cluster, and to design a clustering algorithm for parallel implementation. Thus improving the performance of clustering algorithm to deal with massive data. To sum up, by using K-means algorithm to cluster analysis of power quality indicators of power enterprises, on the one hand, it can clear the development level of the relevant indicators of the power enterprises, it is conducive for enterprises to continuously improve their own shortcomings, on the other hand, through the cluster analysis, the indicators can not only make a comprehensive analysis and comparison of

© Springer Nature Singapore Pte Ltd. 2017
G. Chen et al. (Eds.): PAAP 2017, CCIS 729, pp. 533–540, 2017.
DOI: 10.1007/978-981-10-6442-5_51

enterprise power management, but also can accurately identify the root causes of the gap between the power supply. Hadoop is an open source cloud computing platform that can be used for parallel processing of large-scale data. It has five characteristics, namely reliability, extensible property, high efficiency, fault-tolerant and low cost. MapReduce [1, 8] is a programming model for the parallel operation of large scale data sets. It specifies a map function, to a set of key value mapping into a new set of keys and specify a reduce function to ensure the mapping of all key value pairs in each share the same set of keys. Hadoop [7] can be widely used in large data processing applications due to its own natural advantages in terms of data extraction, deformation and loading. Hadoop distributed architecture, data processing engine as much as possible near the storage of such as ETL such batch operation relatively appropriate, because such operation of batch processing results can be directly to the storage. MapReduce Hadoop function to achieve a single task break, and will be sent to a number of pieces of mission nodes, and then in the form of a single data set to load into the data warehouse. Hadoop distributed architecture makes data processing engine as much as possible near the storage of such as ETL [10] such batch operation relatively appropriate, because such operation of batch processing results can be directly to the storage. Mapreduce function in the Hadoop achieves to break a single task and send a fragment mission to multiple nodes, then load into a data warehouse in the form of a single data set. So it is the right one to deal with the massive data [5].

2 Related Work

Dundar [3] draw a conclusion that K-means based unsupervised feature learning is a powerful alternative to deep-learning algorithms as well as to conventional techniques that rely on handcrafted features. Wu [11] proposed a K-means algorithm based on Sim Hash, which is used to calculate the feature vectors extracted, and then the fingerprint of each text is obtained, to deal with high dimensional and sparse data effectively and greatly reduce the speed of K-means clustering algorithm. Bai [2] proposes a load model based on the K-means algorithm uses the actual operation data of the power network and voltage static characteristic of load is considered. So it can reflect the actual situation of the power load more clearly. In addition, the clustering algorithm is applied to the processing of load data so that the load characteristic data of each time period can be extracted and thins typical methods. Lee [4] proposes a classification method which combines k-means algorithm and Bayesian inference to build a classifier. The proposed method makes the classifier updated as new data are accumulated and in addition adjusted according the concept drift using a windowing mechanism. To handle big data, the proposed method is realized using the map-reduce paradigm which can be deployed in the big data framework. K-means is a popular clustering algorithm to find the clustering easily by iteration. But the computational complexity of the traditional k-means due to accessing the whole data in each cycle of iterative operations is too great to make it fit for very large data set. This paper presents a new clustering algorithm we have developed, fast k-means clustering algorithm based on grid data reduction (GDR-FKM), by which clustering operations can be quickly performed on

very large data set. Application of the algorithm to analysis of the data relativity in TT&C has demonstrated its celerity and accuracy.

From the above researches we can see that only the traditional serial clustering algorithm is studied. But the traditional serial clustering algorithm can not deal with massive data, so we have to study parallel clustering algorithms. In this paper, according to the principle of K-means algorithm and the application of a wide range of MapReduce parallel computing, we propose a parallel implementation of K-means algorithm.

3 Problems and Algorithms

3.1 Power Big Data Mining Problem

Power "big data" concerns power generation, transmission, substation, distribution, electricity and scheduling, it is an analysis and mining of cross-unit, cross-professional and cross-business data. Power big data consists of structured data and unstructured data, with the application of smart grid construction and internet of things, unstructured data is showing a rapid growth momentum, the amount will be larger than that of structured data. The characteristics of power big data meets the five characteristics of large data: Volume, Velocity, Variety, Value and Veracity and there is a high value for the improvement of the profit and control level of electric power enterprises. Big data technology will accelerate the pace of intelligent control of power companies to promote the development of smart grid. For example, we can monitor facilities operation condition dynamically based on the sensor of electric power infrastructure. So that we can effectively change the mode of operation and maintenance. The operation and maintenance fault are eliminated from the bud stage and intelligent operation and maintenance will be achieved.

- **Mapreduce**

The Map/Reduce framework consists of a single job tracker and each cluster node a task tracker. Master is responsible for the scheduling of all tasks that constitute a job, these tasks are distributed on different slaves, master monitors their execution, reimplementation of the failed task. Slave is only responsible for performing tasks assigned by master. The process of Mapreduce is as below. (1) User program in the MapReduce library first divides input file into M blocks (Hadoop default 64M, this parameter can be determined by parameter modification). Then the processing program is executed on the cluster machine. (2) The master control program master assigns the Map task and the Reduce task to the job execution machine worker. A total of Map R tasks and Reduce M tasks need to be assigned. Master will select the free worker and assign these Map tasks or Reduce tasks to the worker node. (3) A Map task assigned by the worker to read and process the relevant input data blocks. From piece of input data parsing out the key, then the key to transfer a user-defined map function, by the map function to generate and output the intermediate key/value key set, these keys to set will be temporarily cached in memory. (4) The key/value in the cache of the partition function is divided into R regions, then periodically written to the local disk. (5) When the

worker Reduce program receives the data store location information sent by the master program, it will use the RPC to read the cached data from the host's keyboard from the worker Map host. After reading all of the intermediate data in worker Reduce, the key is sorted by the same key value of the data together. Because many of the different key values are mapped to the same Reduce task, it is necessary that they should be sorted. (6) Worker Reduce traversal sort of intermediate data, for each of the only intermediate key values, worker Reduce program will send the key value and its associated intermediate value of the collection to the user defined Reduce function. The output of the Reduce function is forced to add to the output file of the partition. (7) Finally, after the successful completion of the task, the output of the MapReduce is stored in R output files.

- **Hadoop HDFS Architecture**

Compared with distributed file system architecture based on P2P model, HDFS uses a distributed file system based on Master/Slave. A HDFS cluster contains a single Master node and a number of Slave node servers. The HSDF architecture design is shown in Fig. 1 below. A single Namenode node greatly simplifies the structure of the system. Namenode is responsible for the preservation and management of all HDFS metadata, so the user data does not need to pass namenode, that is to say the file data is read and write directly on the datanode. HDFS stored files are split into fixed size blocks. When you create a block, the namenode server will assign a unique block identifier to each block. Datanode server store the block in the form of linux files on the local hard disk and read data in accordance with the specified block identifier and byte range. For reliability considerations, each block will be copied to multiple datanode servers. By default, HDFS will use three redundant backups. Of course, users can set different file namespace for different copy number. Nematode manages all file system metadata. These metadata include name space, access control information, block mapping information and current block district information. Nematode also manages the activities within the system, such as block rental management, recovery of isolated block. Nematode makes the information cycle and each datanode server communication and send command to each datanode server and receive block information in datanode.

- **K-means Clustering Algorithm**

K-means algorithm accepts the parameter K; and then the prior input of the N data object is divided into K clustering in order to make the obtained clustering to meet: The similarity of objects in the same cluster is higher, and the similarity of objects in different clusters is small. Cluster similarity calculates the mean value of the objects in each cluster to obtain a "center".

3.2 Realization of Parallel K-means Clustering

- **Design of Map Function**

Map function input is the default format of Mapreduce framework, Key is the offset of the current sample relative to the starting point of the input data file. Value is a string

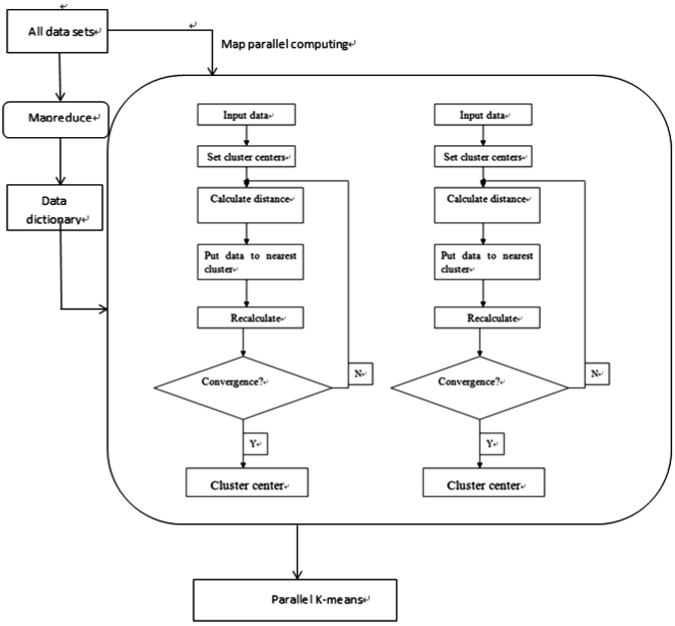

Fig. 1. Parallel realization of K-means algorithm

consisting of the values of each dimension of the current sample. The value of each dimension of the current sample is analyzed from value, then calculate the distance between the center and the K, find the nearest cluster index. The final output is. The parallel realization is shown in Fig. 1 together with pseudo code as follow.

- **Design of Combine Function**

First of all, we parse out the coordinate values of each sample from the list of the string in order. And the corresponding coordinate values of each dimension are added separately. At the same time we record the total number of samples in the list. Pseudo code is shown as follow.

- **Design of Reduce Function**

In the Reduce function, the number of samples processed from each Combine and the coordinates of each dimension of the corresponding nodes are first resolved. Then the corresponding value of each dimension is added separately, and then divided by the total number of samples. That is to get a new center point coordinates. Pseudo code is shown as follow, and the parallel realization of K-means algorithm is shown as follow.

Algorithm 1 K-means algorithm

Input:
K: Numbers of clusters
D: A data set containing N objects
Output: K data sets
(1) **For** K objects in D
(2) **Do**
(3) $C^i - \arg\min \left\| x^i \quad u_j \right\|^2$ //Calculation of the class
(4) Recalculate centroid
(5) Until convergence
(6) **End for**

Algorithm 2 Map Function

Input: <key, value>
Output: <key', value'>
(1) Index is initialized to -1;
(2) **For** i=0 **to** k-1
(3) **Do**
(4) dis=instance;
(5) **if** dis < minDis
(6) minDis = dis; index = i;
(7) **End if**
(8) **End for**
(9) Index is regarded as key';
(10) value=value';
(11) **Output** <key', value'>;

Algorithm 3 Combine Function

Input: <key, value>
Output: <key', value'>
(1) Initialize an array;
(2) Initialize variable num;
(3) **While** (V.hasNext())
(4) **Do**
(5) values in V ;
(6) add values to array.;
(7) num++;
(8) key=key';
(9) We constructs a string that contains information about the num and the various components of the array, which is used as a value';
(10) **Output** <key', value'>;

Algorithm 4 Reduce Function

Input: <key, value>
Output: <key', value'>
(1) Initialize an array;
(2) Initialize V ;
(3) **While** (V.has.Next())
(4) **Do**
(5) values in V ;
(6) add values to array.;
(7) num+=num;
(8) obtain new center.;
(9) key=key';
(10) Construct a string that contains information about the value of each dimension of the new center point, which is used as a value.;
(11) **Output** <key', value'>;

4 Performance Analysis

4.1 Analysis of Time Complexity

We set the amount of the data set is N, the subset size is SN, the number of the clustering is K, the dimension is D, there are P datanodes. The time consists of communication time and computing time. In an iteration, the computing of namenode consists of: (1) The time of dividing clusters:P $*$ SN. (2) The time of receiving clustering result and saving it:SN. (3) The time of dividing remaining data sets:N – P $*$ SN. Datanode is mainly responsible for calculating the distance between every data object and clustering center, the time cost is SN $*$ K $*$ D. The total time is O(SN $*$ K $*$ D $*$ Rp + 2 N $*$ Rp), Rp is the iteration times of parallel algorithm and 2 N $*$ Rp is the communication time.

4.2 Analysis of Space Time Complexity

Given a data set of N, the space that namenode needs is O(N), the space each datanode needs is O(SN), so the total space time complexity of the parallel algorithm is O(N).

4.3 Accelerate Rate Analysis

Let Rs be the iteration times of the serial K-means algorithm, we can calculate the rate is

$$Sp = \frac{O(Rs * K * N * D)}{O(Rp * K * SN * D + 2N * Rp)} = O\left(\frac{1}{\frac{SN}{N} + \frac{2}{K*D}}\right) \quad (1)$$

Because SN is less than N/P, that is to say SN/N is larger than 1/P, so Sp is larger than 1 and is less than P, we can see that the parallel algorithm definitely saves time.

5 Conclusion

This paper mainly introduces the idea of clustering algorithm and parallel computing. The innovation of this paper is proposing the idea of parallel clustering algorithm based on the traditional clustering algorithm, which can effectively deal with the bottleneck of data mining under the current big data environment. However, in this paper, we only propose a framework based on previous studies, we do not realize it, and this is what we will do in the future work.

Acknowledgment. This work is supported by National Science Foundation of China Grant #61672088, Fundamental Research Funds for the Central Universities #2016JBM022 and #2015ZBJ007, Open Research Funds of Guangdong Key Laboratory of Popular High Performance Computers. The corresponding author is Yidong Li.

References

1. Aragues, R., Sander, C., Oliva, B.: Predicting cancer involvement of genes from heterogeneous data. BMC Bioinform. **9**(1), 1–18 (2008)
2. Bai, Z.G., Zhang, H.D.: k-means clustgering algorithm based on mutation. J. Anhui Univ. Technol. **4**, 019 (2008)
3. Dundar, M., Kou, Q., Zhang, B., He, Y.: Simplicity of kmeans versus deepness of deep learning: a case of unsupervised feature learning with limited data. In: IEEE International Conference on Machine Learning Applications (2015)
4. Lee, K.M.: Grid-based single pass classification for mixed big data. Adv. Nat. Appl. Sci. **9** (21), 8737–8746 (2014)
5. Monmarch, N., Slimane, M., Venturini, G.: AntClass: discovery of clusters in numeric data by an hybridization of an ant colony with the Kmeans algorithm (2003)
6. Naimi, A.I., Westreich, D.J.: Big data: a revolution that will transform how we live, work, and think. Information **17**(1), 181–183 (2014)
7. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The hadoop distributed file system. In: IEEE Symposium on MASS Storage Systems and Technologies, pp. 1–10 (2010)
8. Triguero, I., Peralta, D., Bacardit, J., García, S., Herrera, F.: MRPR: a MapReduce solution for prototype reduction in big data classification. Neurocomputing **150**(150), 331–345 (2015)
9. Varian, H.R.: Big data: new tricks for econometrics. J. Econ. Perspect. **28**(2), 3–28 (2014)
10. Vassiliadis, P., Simitsis, A., Skiadopoulos, S.: Conceptual modeling for ETL processes. In: ACM International Workshop on Data Warehousing and Olap, pp. 14–21 (2002)
11. Wu, G., Lin, H., Fu, E., Wang, L.: An improved k-means algorithm for document clustering. In: International Conference on Computer Science and Mechanical Automation, pp. 65–69 (2015)