TELKOMNIKA, Vol.14, No.3, September 2016, pp. 1123~1127
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v14i3.3771

MapReduce Integrated Multi-algorithm for HPC Running State Analysis

ShuRen Liu*, ChaoMin Feng, HongWu Luo, Ling Wen
Northwest Branch of PetroChina Research Institute of Petroleum Exploration and Development,
Lanzhou 730020, China
*Corresponding author, e-mail: 1648937949@qq.com

Abstract

High-performance computer clusters are major seismic processing platforms in the oil industry and have a frequent occurrence of failures. In this study, K-means and the Naive Bayes algorithm were programmed into MapReduce and run on Hadoop. The accumulated high-performance computer cluster running status data were first clustered by K-means, and then the results were used for Naive Bayes training. Finally, the test data were discriminated for the knowledge base and equipment failure. Experiments indicate that K-means returned good results, the Naive Bayes algorithm had a high rate of discrimination, and the multi-algorithm used in MapReduce achieved an intelligent prediction mechanism.

Keywords*: high-performance clusters (hpc), hadoop, mapreduce, k-means, naive bayes*

1. Introduction

Seismic processing technology is one of the primary means for oil and gas exploration and development. At present, high-performance computer clusters are major seismic processing platforms in the oil industry. However, the cluster sizes are expanding with increasing amounts of data processing; meanwhile, various software applications are being used interchangeably, leading to frequent cluster failures. Therefore, stability factors have become increasingly important. Here, an intelligent prediction mechanism is introduced to build a knowledge base from historical data and detect hidden faults in the cluster using data mining techniques before the maintenance node crashes. This method will minimize node failure impacts on oil and gas exploration projects.

Hadoop is an open-source cloud computing model [1] that uses MapReduce [2] for the parallel computation of big data. Owing to its high reliability, data processing capacity, flexibility, and scalability, this model has gradually become popular for computer research and is widely used by search engines, machine learning and so on [3-5]. However, Hadoop has not yet been used to monitor high-performance cluster running conditions.

Together, Hadoop and MapReduce make intelligent prediction mechanisms possible for high-performance cluster running analyses. Related work has been carried out on k-means [6] and Bayesian [7] Mapreduce parallelization improvements, but no comprehensive use, in this paper, multi-algorithm was applied for HPC running state analysis.

2. Research Method

IN the architecture, the entire cluster is described by each Linux system state quantity component, which characterizes the cluster state. A state data analysis platform was built based on Hadoop platform characteristics and high-performance cluster system status data. The platform comprises three parts (Figure 1):

a) A state collection module collecting the high-performance cluster running status data.
b) A state data storage module that uses HBase to efficiently achieve huge dynamic timing of the historical status data.
c) A data analysis module, the core content of this article, that includes two algorithms based on MapReduce and the K-Means and Naive Bayes algorithms.

Received January 10, 2016; Revised April 20, 2016; Accepted May 8, 2016

Linux commands are used for clusters running the state data collection, which are embedded in a Java program. After the acquisition is complete, the HBase API interface is called to store the data. Running status characteristics are divided into health, general, and fault, and different categories are refined as the knowledge base expands.

2.1. Implementation of the K-means Algorithm in MapReduce

The K-means [8] algorithm uses distance as the similarity evaluation index and outputs the k cluster centers (Figure 2). The steps of the implementation process are described below.

a) K data centers are selected from the data set.
b) All of the data are used to measure the distance between each center to find the minimum distance, which is included in the minimum class.
c) All types of centers are recalculated. Steps 2 and 3 are repeated until the threshold is met. The main function for an appropriate threshold design is to use an iterative process to achieve the Map and Reduce functions and to continue to function calls until the threshold is met.

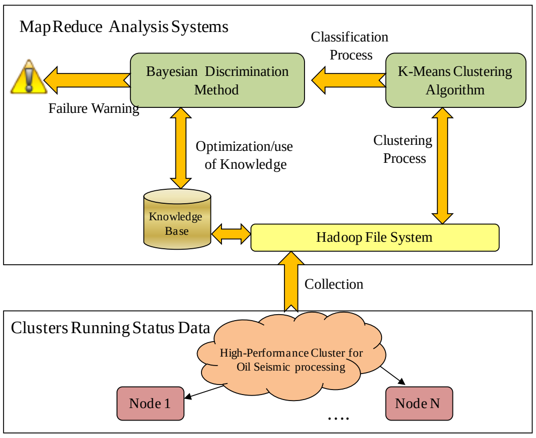


Figure 1. The architecture of the MapReduce Integrated Multi-algorithm

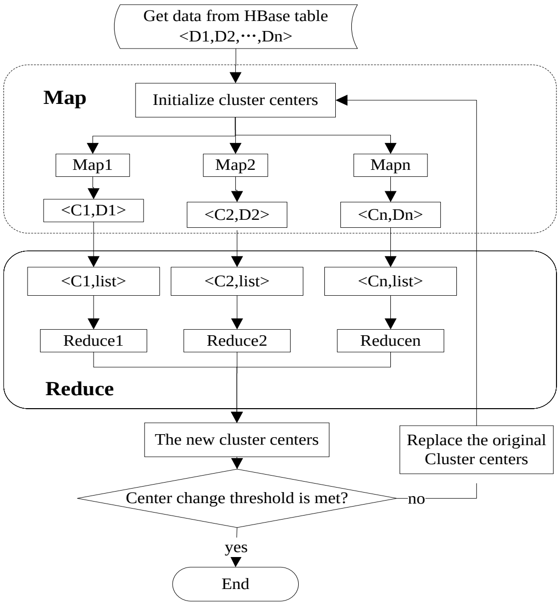


Figure 2. The Implementation Process of the K-means Algorithm in MapReduce

2.2. Implementation of the Naive Bayes algorithm in MapReduce

Figure 3 shows the implementation process for the Naive Bayes algorithm [9-10] in MapReduce, which is set up as follows:

a) Let $X = \{a_1, a_2, \dots, a_m\}$ for an item to be classified, where each a is a characteristic property of X.
b) Set $C = \{y_1, y_2, \dots, y_m\}$, where each y is a category.
c) Calculate $P(y_1|x)$, $P(y_2|x) \dots P(y_n|x)$.
d) If $P(y_k|x) = \max\{\ P(y_1|x), P(y_2|x), \dots, P(y_n|x)\ \}$, then $x \in y_k$.

The key is how to calculate the probability of each condition in Step 3 by obtaining a known item classification called the training set. Conditional probability estimates of each characteristic property in each category are counted using Equation 1:

$$\begin{cases} P(a_1|y_1), P(a_2|y_1), \ldots, P(a_m|y_1) \\ \ldots \\ P(a_1|y_n), P(a_2|y_n), \ldots, P(a_m|y_n) \end{cases} \tag{1}$$

If the property of each characteristic condition is independent, they can be calculated using Bayes' theorem:

$$P(y_1|x) = \frac{P(x|y_i)P(y_i)}{P(x)}. \tag{2}$$

Because the denominator is a constant for all categories, we need to maximize each component. Each attribute has conditional independence, therefore:

$$P(x|y_i)P(y_i) = P(a_1|y_i)P(a_2|y_i) \ldots P(a_m|y_i)P(y_i) = P(y_i)\prod_{j=1}^{m} P(a_j|y_i). \tag{3}$$

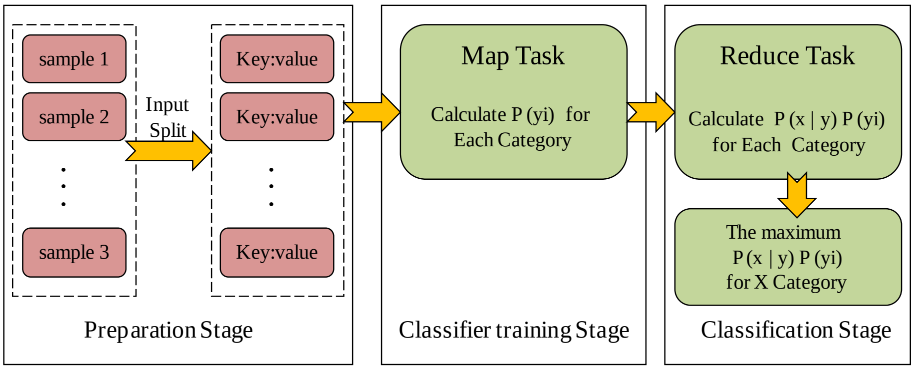


Figure 3. The implementation process for the Naive Bayes algorithm in MapReduce

3. Results and Analysis

On five BCL460c blades, a fully distributed mode Hadoop platform was built, including a namenode and four datanodes. Each node had a 10-core CPU, 64 GB RAM, and a 600 GB hard drive. The operating system was RedHat 5.8, and we used jdk1.7.0_25, Hadoop version 2.5.0, and HBase version 0.98.1. The monitored object was a high-performance cluster with 512 nodes.

3.1. K-means Results

Stand-alone K-means (Matlab program, Intel i5, for 128 GB memory) and Hadoop were run and property item 20 of the 10,000 running status data is shown in Figure 4. It can be seen from Figure 4 that Hadoop runs faster than the stand-alone mode when the number of iterations increases.

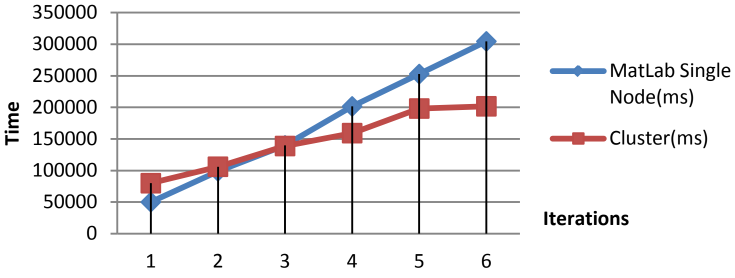


Figure 4. The Relationship between the Iteration and the Running Time for Stand-Alone K-means (blue) and Hadoop (red)

Figure 5 shows results by Hadoop on 10,000 running status data iterms with 10, 20 and 30 properties, and then the absolute distance values of the three cluster centers were compared. The more attribute items, the better the clustering results.

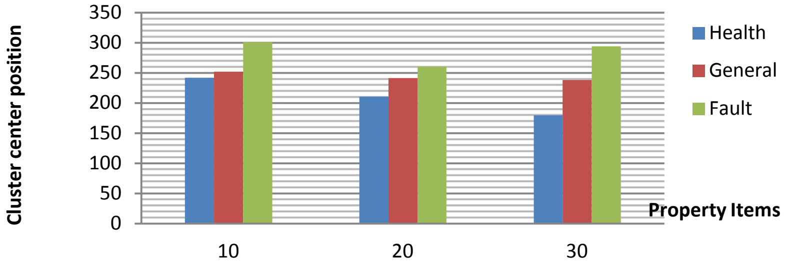


Figure 5. Results for Different Attributes

The experiments indicate that the running time of the MapReduce program is shorter than that of the stand-alone program in the case of a larger number of iterations and that the more iterations and attribute items there are, the better the K-means clustering results.

3.2. Naive Bayes Classifier Results

Figure 6 demonstrates that the collected high-performance computer cluster running status data were first clustered by K-means, and then the results were used for Naive Bayes training. Finally, the test data were discriminated for the knowledge base and equipment failure.

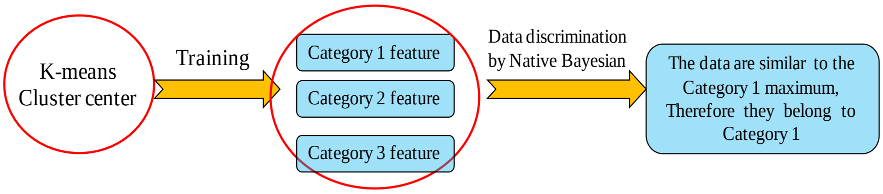
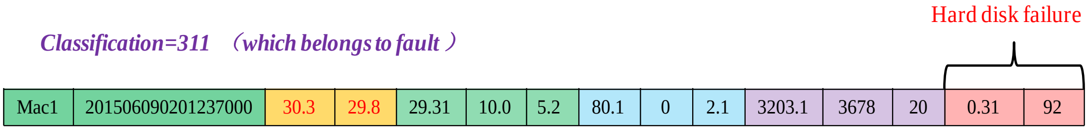


Figure 6. Process Flow of the HPC Running State

Figure 7 shows an example for determining a fault. Running data were collected from a computer on June 9, 2015, which was given the classification 311, where 311 belongs to the fault classification. In actuality, this computer experienced a hard drive failure. Therefore, this classification/discrimination was appropriate.



Classification=311 （which belongs to fault ）

Hard disk failure

Mac1	201506090201237000	30.3	29.8	29.31	10.0	5.2	80.1	0	2.1	3203.1	3678	20	0.31	92

Figure 7. An Example for Determining A Fault

In Figure 8, a cross plot shows the discrimination results for different attributes, running K-means on Hadoop for a maximum of 10 iterations 500 times for attributes from 5 to 30 by 10,000 status data. If each cluster center is used as a sample, then six knowledge bases are generated from the 500 samples. Each knowledge base was trained by means of Naive Bayes based on MapReduce and compared with a single node using 10,000 raw data for the 100 test data classifications.

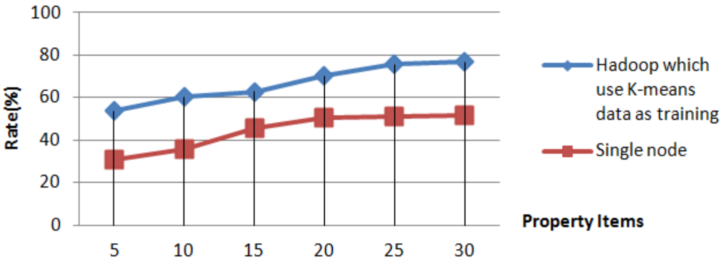


Figure 8. Discrimination Results for Different Attributes.

The experiment that used K-Means intermediate data as training data performed better than the traditional method on a single mode. As the property items increased, the discrimination success rate increased. Owing to the number of samples (with a maximum of 10,000), the sample attribute items (up to 30 uses), and the impact of a possible correlation between the properties of the items, the success rate was below 80%; however, it is practical for monitoring high-performance clusters.

4. Conclusion

To enhance the stability of high-performance clusters in oil and gas exploration, Hadoop was used to analyze the high-performance cluster running state. Analysis for equipment failure was achieved via K-Means and Naive Bayes algorithms programmed into MapReduce. Experiments indicate that K-means returned good results, the Naive Bayes algorithm had a high rate of discrimination, and the multi-algorithm that used MapReduce achieved an intelligent prediction mechanism.

References

[1] Addair TG, Dodge DA, Walter WR, et al. Large-scale seismic signal analysis with Hadoop. *Computers & Geosciences*. 2014; 66(2):145-154.
[2] Dean J, Ghemawat S. *MapReduce: Simplified Data Processing on Large Clusters*. In Proceedings of Operating Systems Design and Implementation. 2004; 51(1): 107-113.
[3] Londhe S, Mahajan S. Effective and Efficient Way of Reduce Dependency on Dataset with the Help of Mapreduce on Big Data. *Telkomnika Indonesian Journal of Electrical Engineering*. 2015; 15(1).
[4] Jayalath C, Stephen J, Eugster P. From the Cloud to the Atmosphere: Running MapReduce across Data Centers. *Computers IEEE Transactions on*. 2014; 63(1): 74-87.
[5] Liu Y, Wei W, Zhang Y. Checkpoint and Replication Oriented Fault Tolerant Mechanism for MapReduce Framework. *Telkomnika Indonesian Journal of Electrical Engineering*. 2014; 12(2).
[6] Aljarah I, Ludwig SA. *Parallel Glowworm Swarm Optimization Clustering Algorithm based on MapReduce*. IEEE Symposium Series on Computational Intelligence. 2014: 1-8.
[7] Villa S, Rossetti M. Learning Continuous Time Bayesian Network Classifiers Using MapReduce. *Journal of Statistical Software*. 2014; 62(3): 1-25.
[8] Kanungo T, Mount DM, Netanyahu NS, et al. An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*. 2002; 24(7): 881-892.
[9] Ahmed SE. Bayesian Networks and Decision Graphs. *Technometrics*. 2002; 50(1): 362.
[10] RRding KP, Wolpert DM. Bayesian integration in sensorimotor learning. *Nature*. 2004; 427(6971): 244-247.