IJCAT International Journal of Computing and Technology, Volume 1, Issue 3, April 2014
ISSN : 2348 - 6090
www.IJCAT.org

Parallel K-Means Clustering Based on Hadoop and Hama

[1] Ashish A. Golghate, [2] Shailendra W. Shende

[1] Department of Information Technology, Yeshwantrao Chavan College of Engineering, Nagpur, Maharashtra, India.

[2] Department of Information Technology, Yeshwantrao Chavan College of Engineering, Nagpur, Maharashtra, India

Abstract - The drawbacks of Map-Reduce are simplifying reliability and fault tolerance, MR jobs does not preserve data in memory. In the distributed file system MR jobs dump does not read next MR Job. The Bulk Synchronous Parallelism does not suffer this drawbacks, it is alternative to the MR model. Clustering techniques is used for data analysis across all disciplines. K-means is clustering algorithm its simplicity and speed to run on large data set. The performance of K-means is inefficient when large data set will be used. Map-Reduce and Bulk synchronization Parallelism solve the problem of inefficiency when large data set used. The experiments show that our approach is efficient and Bulk synchronization Parallelism is efficient as compare to the Map-Reduce.

Keywords **- Map-Reduce, Bulk synchronization Parallelism, K-means, Clustering.**

1. Introduction

Clustering is important unsupervised learning method. A definition of clustering is "the process of organizing objects into groups whose members are similar in some way" [4].Cluster is set of objects is similar between them and dissimilar to the other clusters objects. K-means is one of the very fast clustering algorithm and one of the top 10 algorithms in the data mining. The K-means have several problems; the main problem is dealing with high dimension and large data set. Map-Reduce and Bulk synchronization Parallelism is programming model for implementation for generating and processing large data set. To solve the problem mention above, Map-Reduce used the Hadoop programming model and Bulk synchronization Parallelism is used HAMA programming model to solve the inefficient problem in clustering on large data sets. Map-Reduce (MR) programming model is popular for large-scale data analysis and Hadoop is used for the implementation of this MR programming model on very large scale. The major disadvantage of the MR model is simplify reliability and fault tolerance, it does not store data in memory between map and reduce tasks of MR jab [5]. The implementation of the MR model , the data is pass to the next MR job, in the distributed file system(DFS) the MR job must dump its data, its does not read the next MR job. To overcome this problem we used the Bulk synchronization Parallelism is it allows to run algorithms entirely in memory of the cluster.

Bulk synchronization Parallelism is programming model consists of a sequence of supersteps. Each BSP superstep is implementation in parallel by every peer in the BSP computation. The superstep consists of a local computation, a process communication, and barrier synchronization [13]. For BSP computation we used Apache's Hama platform built on the top of Hadoop that can be used process any kind of data in the collective memory of the cluster.

The main contribution of this paper is summarized as follows:

The inefficiency problem in clustering on large data sets is solved by using a distributed computing framework Map Reduce and Bulk Synchronous Parallelism.

2. Related Work

In 2004 Google was first introduced the Map-Reduce (MR) model. Much large-scale organization used this MR model, Hadoop is one of the most popular MR model. Hadoop is open-source project developed by Apache [2]. To perform data analysis now days Yahoo and many other companies used the Hadoop for the implementation. In 1990 Leslie G. Valiant introduced the BSP model and

then since improved and extended by many others. The best known implementations of the BSP model for data analysis on the cloud are Hama. The BSP Hama it has not reached the popularity of MR yet. Its open-source counterpart, Giraph, and the general BSP platform, Hama, are still in their very early stages of development and are not used in the same degree as Hadoop

3. Background

In this section, introductions about the algorithms and technologies adopted in this paper are given. An outline on k-means is given in section 1, Map Reduce in section 2 and Bulk Synchronous Parallelism in section 3.

3.1 K-means

K-means first used in 1967 by James MaeQueen, the K-means idea can be traced by Steinhaus in 1956.K-means is an algorithm to classify or to group your objects based on features into K number of group. The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid. The purpose of K-mean clustering is to classify the data. The K-means is cluster analysis method which objective is to partition n object in to k clusters; in this each object belongs to the nearest centroid of the cluster.

Algorithm description as following:

Input: The number of clusters k and n documents
Output: k clusters

1. Randomly select k documents from n documents as the initial cluster centers.
2. Calculate the distances of the rest documents to the very center of the clusters, and assign each of the rest documents to the nearest cluster.
3. Calculate and adjust each cluster center.
4. Iterate Step2 ~ Step3 until the criterion function converge.
The program ends.

3.2 Map Reduce

In distributed data storage we need to consider such as synchronization, concurrency, and load balancing for the underlying system .In this simple calculation become very complex. Map Reduce is a programming model implementation for processing an generating large data sets, which was introduced by Google to solve the distributable problems. This Map Reduce programming model solves the problems such as data distribution, fault tolerance, machine to machine communication.

3.3 Map Reduce Programming Model

Map Reduce programming model map and reduce function used the Mapper and Reducer interfaces.

3.3.1 Mapper

Map function requires handling the input of a pair of key value and produces a group of intermediate key and value pairs. <key,value> consists of two parts, value used for the data related to the task, key stands for the "group number " of the value . Map Reduce combine the intermediate values with same key and then send them to reduce function.

3.3.2 Reducer

Reduce function is also provided the intermediate key pairs and the value set relevant to the intermediate key value. Reduce function mergers values, to get a small set of values. The process is called "merge ". There are complex operations in the process. Reducer makes a group of intermediate values set that associated with the same key smaller.

3.3.3 K-means with Map Reduce

The k-means algorithm spends most execution time on calculating the distances between objects and cluster centroids. This process is, however, necessary for the algorithm itself, and thus we must consider about improving the efficiency of the algorithm from other aspects. Therefore, enhancing the performance of the distance calculation is the key to improve the time performance of the algorithm. It is easy to note that the execution orders of distance calculation of objects will not affect on the final result of clustering. Therefore, the distance-calculating process can be executed in parallel by using the Map Reduce framework.

The maper function of the k-means algorithm using Map Reduce is shown follows:

Input: centroids, input.
Output: output.
1: nstCentroid ← null, nstDist ← ∞
2: for each c ∈ centroids do
3: dist ← Distance (input. value, c);
4: if nstCentroid == null || dist < nstDist then

5: nstCentroid← c, nstDist ← dist;
6: end if
7: end for
8: output. collect(nstCentroid, object);

The reducer function of the k-means algorithm using Map Reduce is shown follows:
Input: input.
Output: output.
1: v ← Φ ;
2: for each obj ∈ input. value do
3: v ← v ∪ {obj};
4: end for
5: centroid ← ReCalCentroid(v);
6: output. collect(input. key, centroid);

3.4 Bulk Synchronous Parallelism

Bulk Synchronous Parallelism computer consists of processors connected by communication network. A BSP computation proceeds in a series of global supersteps. The superstep consists of a local computation, a process communication, and barrier synchronization.
HAMA is a distributed computing framework based on BSP (Bulk Synchronous Parallel) computing technique for massive scientific computations.

3.4.1 K-means with Apache Hama's BSP

In a typical clustering scenario you will have much more points than centers/means. So n >> k where n is number of vectors and k is number of centers.

Since Apache Hama 0.4.0 will provide us with a new I/O system it makes it really easy to iterate over a chunk of the input on disk over and over again, we use this fact and put all centers into RAM.

The trick in this case is that unlike in graph algorithms, we not split the centers over the tasks, but every task holds all k-centers in memory. So each task gets a part of the big input file and every task has all centers. Now we easily do assignment step, we just iterate over all input vectors and measure the distance against every center.
While iterating we find the nearest center for each of the n vectors. To save memory we are going to average our new center "on-the-fly" At the end of the assignment step, we have in each task the "on-the-fly" computed average new centers. Now broadcast each of this computed averages to the other tasks.

Then we are going to sync so all messages can be delivered. Afterwards we are iterating in each task through the messages and averaging all incoming centers if they belong to the same "old" mean.

The fig.1 shows it is about the exchange of the locally computed mean for two tasks.

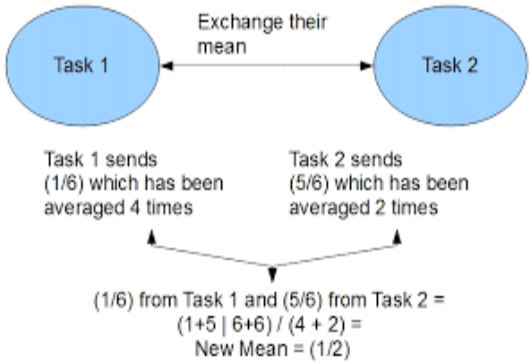

Fig.1 Exchange of the locally computed mean for two tasks

As we can see on this fig, we have two tasks which have calculated "their version" of a new mean. Since this isn't the "global mean" we have to calculate a new mean that will be consistent across all the tasks and is still the correct mathematical-mean.

Apply the Average on data streams strategy. Each task is going to get the local computed averages from each other task and is reconstructing the global mean.

Actually this is the whole intuition behind it. As the algorithm moves toward, this whole computation is running all over again until the centers converged = don't change any more. This is much faster than Map Reduce, because we don't have to submit a new job for a new computation step. In BSP superstep is less costly than running a Map Reduce job is faster.

4. Experimental Environment and Data Set

In this paper, experiments are based on a PC with the following hardware configuration: Intel (R) Core (TM) DuoCPU T6570 @ 2.10GHZ, 2.10GHZ, 2.00GB RAM and250GB hard disk. The software environment uses the same configuration: Linux operating system; the distributed computing platform of Hadoop and Hama; and Java development platform JDK 1.6. The data used in this experiment comes from text classification corpus of 20_newsgroups Data Set.

Experiment 1 Executing time by K-Means algorithm on stand-alone computer and based on Map Reduce and Hadoop for the same number of documents.

1.

Clustering	*K-Means clustering*
The number of Documents (Size)	6000 (21 MB)
Number of Iterations	50
Execution time	1:28:36:72

2.

Clustering	*K-Means clustering*
The number of Documents (Size)	4000(17 MB)
Number of Iterations	30
Execution time	00:32:00:83

Experiment 2. Executing time by K-Means algorithm on stand-alone computer and based on BSP and HAMA for the same number of documents.

1.

Clustering	K-Means clustering
The number of Documents (Size)	6000 (21 MB)
Number of Iterations	50
Execution time	1:02:30:6

2.

Clustering	K-Means clustering
The number of Documents (Size)	4000(17 MB)
Number of Iterations	30
Execution time	00:22:05:63

5. Conclusion

In this paper, work represents only a small first step in using the Map Reduce programming technique in the process of large-scale data Sets. We take the advantage of the parallelism of Map Reduce to design a parallel K-Means clustering algorithm based on Map Reduce. This algorithm can automatically cluster the massive data, making full use of the Hadoop cluster performance. It can finish the text clustering in a relatively short period of time. But Bulk Synchronous Parallelism is much faster than Map Reduce, because you don't have to submit a new job for a new computation step. In BSP the superstep is less costly than running a Map Reduce job, therefore it is faster.

References

[1] Aditya B. Patel, Manashvi Birla, Ushma Nair, " Addressing Big Data Problem Using Hadoop and Map Reduce", NIRMA UNIVERSITY INTERNATIONAL CONFERENCE ON ENGINEERING, NUiCONE-2012,.

[2] Tomasz Kajdanowicz, Wojciech Indyk, Przemyslaw Kazienko and Jakub Kukul, "Comparison of the Efficiency of MapReduce and Bulk Synchronous Parallel Approaches to Large Network Processing", IEEE 12th International Conference on Data Mining Workshops 2012

[3] Wadoud Bousdira, Frédéric Gavay, Louis Gesbertz, Frédéric Loulerguex, and Guillaume Petiot, "Functional Parallel Programming with Revised Bulk Synchronous Parallel ML" First International Conference on Networking and Computing 2010

[4] I. Gourlay, P. M. Dew, K. Djemame "Bulk Synchronous Parallel Computing Using a High Bandwidth Optical Interconnect" Proceedings of the International Parallel and Distributed Processing Symposium 2010

[5] Songchang Jin, Shuqiang Yang, Yan Jia "Optimization of Task Assignment Strategy for Map-Reduce" 2nd International Conference on Computer Science and Network Technology 2012

[6] Foto N. Afrati ,Dimitris Fotakis , Jeffrey D. Ullman "Enumerating Subgraph Instances Using Map-Reduce" in WSDM, 2012.

[7] Jiamin Lu, Ralf Hartmut G¨uting," Parallel Secondo: Boosting Database Engines with Hadoop" IEEE 18th International Conference on Parallel and Distributed Systems 2012

[8] F. Afrati and J. Ullman, "Optimizing multiway joins in a map-reduce environment," IEEE Transaction of Knowledge and Data Engineering, vol. 23, no. 9, pp. 1282–1298, 2011.

[9] Hadoop. http://hadoop.apache.org/.

[10] D. Battre, *et al*. Nephele/PACTs: "A Programming Model and Execution Framework for Web-Scale Analytical Processing". In *SOCC'10*.

[11] Hama. http://incubator.apache.org/hama/.

[12] M. Zaharia, *et al*. Resilient Distributed Datasets: "A Fault-Tolerant Abstraction for In Memory Cluster Computing". 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2012.

[13] A. Tiskin. "The Bulk-Synchronous Parallel Random Access Machine". *Theoretical Computer Science, 196(1,2)*, 1998

[14] Hadoop Website, http://hadoop.apac

[15] Zhao, W., Ma, H., et al.: Parallel K-Means Clustering Based on MapReduce. In: CloudCom 2009. LNCS, vol. 5931, pp. 674–679 (2009)

[16] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2000.

[17] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D.Piatko, R. Silverman, and A. Y. Wu, "An Efficient Kmeans Clustering Algorithm: Analysis and Implementation", IEEE Transactions on Pattern Analysis and Machine Intelligence, July, 2002, Vol. 24, No. 7, pp. 881-892.

[18] J. Dean, S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Communications of the ACM, January, 2008, Vol. 51 No. 1, pp. 107-113.

[19] D. Borthakur, "The Hadoop Distributed File System: Architecture and Design", http://hadoop.apache.org/.

[20] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters.In *Proceedings of the 6th USENIX Symposium on Operating Systems Design and Implementation*,pages 137-149, 2004.

First Author

Ashish A. Golghate received the B.E. degree in Information Technology from HVPM Amaravati in 2011 and perusing MTech. degree in Information Technology from YCCE, Nagpur in 2013. He worked as a Teaching Assistant at 2-WEEK ISTE DBMS-WORKSHOP conducted by IIT Bombay (21st may to 31st may 2013) in YCCE Nagpur.

Second Author

Shailendra W. Shende pursuing Phd. from VNIT, Nagpur in the area of Parallel Processing. He worked as a Coordinator at 2-WEEK ISTE DBMS-WORKSHOP conducted by IIT Bombay (21st may to 31st may 2013) in YCCE Nagpur. He now currently working as a Asst. Prof. in Information Technology Department at YCCE Nagpur.