International Journal of Latest Trends in Engineering and Technology
Special Issue SACAIM 2016, pp. 415-421
e-ISSN:2278-621X

A STUDY ON NEW DATA MINING ALGORITHM BASED ON MAPREDUCE AND HADOOP

Melbert D'Cunha[1], Sanford Fernandes[2] and Mithun Dsouza[3] and S Sathyanarayana[4]

Abstract- The goal of data mining is to determine concealed valuable information in large databases. Mining patterns from databases is one of the major issues in data mining. As the database size increases, the computational time and required memory increases too. Based on this, we proposed the MapReduce programming mode which consists of parallel processing to analyze the larger-scale network. All the experiments were done under hadoop, deployed on the cluster which consists of better servers. Through observed evaluations in the various simulation conditions, the proposed algorithms are shown to deliver better performance regarding scalability and execution time.

Keywords – MapReduce, Hadoop, Data mining,Newman algorithm.

I. INTRODUCTION

Business intelligence and data warehouse can manage TeraByte level data or even higher level. Although many methods have been put forward to deal with high-end data, but the query process is a bottleneck [1]. The introducton of cloud computing to the large amount of data mining, Hadoop is a MapReduce programming model and mass data [2]. It has performed a lot of simulation system in the cloud computing, such ascomputing based on the aspects of cloud modeling and simulation platform of COSIM-CSP system [3], a new way of the networked manufacturing [4], cloud framework for visual simulation [5],and the military training system [6]. A simple MapReduce is done by McCreadie on the Hadoop [7].Ralf came up with a basic designed to support cloud computing [8]. Moretti introduced an efficient data mining method, the data and computation is distributed to a cloud

[1] *Department of MCA, Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*
[2] *Department of MCA, Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*
[3] *Department of MCA, Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*
[4] *Department of MCA, Aloysius Institute of Management & Information Technology(AIMIT), St.Aloysius College(Autonomous) Mangaluru, Karnataka - 575022*

computing [9]. Gillick introduced the inquiry learning with Hadoop[10]. Almost all of the data mining algorithms are based on object oriented programmings (OOP) that are usually run on a single system.However, the aspects of the MapReduce mode is not perfectly suitable for data mining.

In order to solve the limitations and problems, this paper uses a Newman parallel search algorithm based on MapReduce, to improve the processing velocity of large data. The test proves that the parallel algorithm is efficient to large data sets in this paper.

II.PROPOSED ALGORITHM

A. *The Newman Algorithm with Modularity –*

The fast Newman algorithm is a clustering algorithm based on greedy algorithm. Its steps are as follows:

Step 1: Initialize the network for N communities. It means that every point is an independent association. The initial element e_{ij} and a_i is satisfies the following formula:

$$e_{ij} = \begin{cases} \frac{1}{2m} \\ 0 \end{cases}$$

When the side is connected between i and j, the $e_{ij} = \frac{1}{2m}$. When the side is not connected between i and j, the $e_{ij}=0$.

$$a_i = \frac{k_i}{2m}$$

Where the $k_i$is the degree of node; m is the total number of edges in the network.

Step 2: Merge the communities successively which is connected by the side, and calculation theincrement of module.

$$\Delta Q = e_{ij} + e_{ji} - 2a_i a_j = 2(e_{ij} - a_i a_j)$$

According to the principle of greedy algorithm, each time with the Qshould be increasedmost or reduce the minimum direction. After the merger by each time, we will update the corresponding elements of e_{ij}, and add the row and column which is correspond tocommunity.

Step 3: Repeat step 2, constantly consolidated community, until the Qvalue is no longerincreases. This has been the best network community structure.

Due to the time complexity of the algorithm is $O((m+n)n)$, so when the data quantity exceeds 10000, the required memory for $1 * 1 0^8$, then the memory overflow.

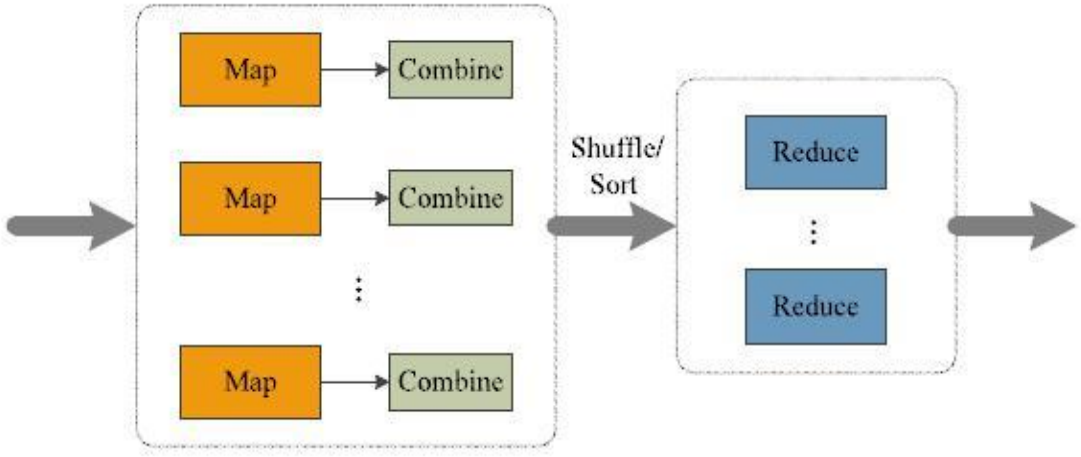

Figure 1. The MapReduce Model

B. The Newman Parallel Algorithm with Modularity –

First initialize
MAP stage
Read the side information

$$Emit(<key_1, key_2>, value)$$

$key_1$and $key_2$is the Edge between vertices, $value$ is the number of the edge.
Then, we will repeat the following procedure:

Step 1:
the edge sequence files will be converted into the degree sequence file Map：
$if(key_1 \square\square\square key_2)$

$Emit(key_1, value)$ $else$
$Emit(key_1, value)$ $Emit(key_2, value)$;
}
Reduce：
$sum \square 0$;
$for(IntWritable value : values)$ $sum \square\square\square value.get$
();
$SumValue.set(sum)$;
$Emit(key, SumValue)$;
Step 2:

Calculated the value of $\square Q$
Map：
Read the edge sequential file
$Emit(\square key_1, key_2 \square\square\square value, (), ())\square$;
Reduce:
Read the degree sequential file
$Emit(\square key, \square 1\square\square\square(), value, ())\square$; $weight \square 0$;
$degree_1 \square 0$; $degree_2 \square 0$;
$\square Q \square 0$;
$for(ValuePair value : values)$

```
{
  if(value.getFirst!□0)
    weight□value.getFirst;
  elseif(value.getSecond!□0)
   {
      if(degree1□□□0)
      degree1□value.getSecond; else
         degree2□value.getThird;
    }
                /        □Q□weightedge□(degree1edge)*(degree2edge); 2
}
Emit(□key1,key2□,□Q);
```

Step 3:

Find out the value of δQwhich has the most vertices

Reduce:

```
intmaxValue=0
while(value.hasNext(){
maxValue=Math.max(maxValue,values.next().get()
)
}
emit(<key1,key2>,max
Value)if(maxValue
==0)
end
```

Repeat the Step 1 to Step 3, until the X value is no longer increases.

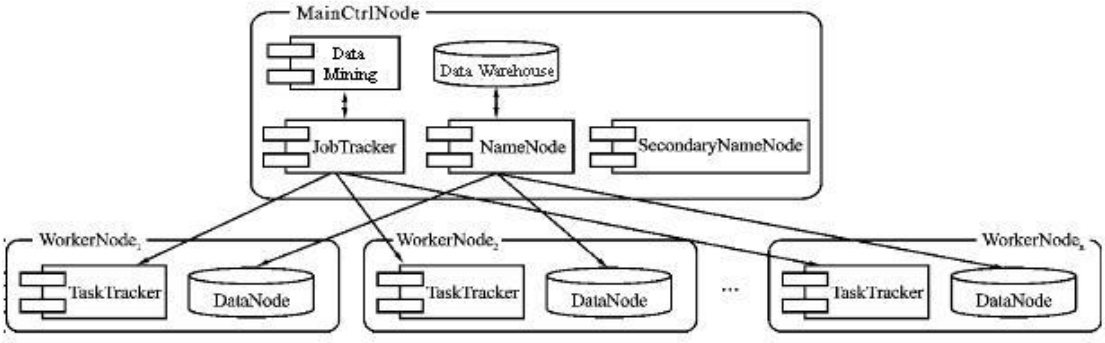

Figure 2. Parallel Data Processing Platform Based on the MapReduce

III. EXPERIMENT AND RESULT

The performance of parallel computing can be measured by its processing speed and scalability. [18-20]. The speedup is the performance improvement that is gained by theparallel computing to reduce the running time. It is an important index to verify the performance of parallel computing. The formula is Sp = Ts /Tp, Where the Ts denotes the

computation time of serial algorithm (*i.e.*, in a single node), and Tp denotes calculation time of parallel algorithm (*i.e.*, in the same p node).The acceleration is larger, the consumption of relative time of parallel computing is small, and the parallel efficiency and performance improvement is bigger.

We continuously give four tables and four graphs as below.

They are the test results in the different number of nodes of the four algorithms.

Table 1. Run Times for the Newman in MapReduce

Unknowns	24	500	1000	2000	4000	6000	8000
1 node	260	263	335	645	1955	3752	7561
2 nodes	256	260	287	505	1352	2489	4425
4 nodes	255	245	291	358	778	1403	2358
8 nodes	249	252	288	387	569	856	1562
16 nodes	229	238	268	300	358	568	954

Table 2. Run Times for the PAM in MapReduce

Objects	10000	25000	50000	750000	100000
1 node	1425	1466	2125	3785	7003
2 nodes	1154	1852	1995	2158	6251
4 nodes	811	793	1258	2368	2685
8 nodes	635	611	1325	1157	1774
16 nodes	288	512	427	788	1121

Table 3. Run Times for the CLARA1 in MapReduce

Objects (thousand)	25	50	100	500	1000	5000	10000
1 node	125	126	138	177	275	882	1658
2 nodes	80	88	90	153	231	512	814
4 nodes	60	66	72	125	133	342	475
8 nodes	52	66	61	124	132	253	325
16 nodes	44	50	58	100	102	121	168

Table 4. Run Times for the CLARA2 in MapReduce

Objects (thousand)	25	50	100	500	1000	5000	10000
1 node	112	123	134	172	276	882	1658
2 nodes	80	88	95	153	235	518	817
4 nodes	60	66	72	124	134	343	476
8 nodes	53	65	67	125	132	253	324
16 nodes	45	51	57	98	99	115	151

IV.CONCLUSION

Regardless of the issues encountered, all implemented algorithms were able to achieve speedup from using multiple nodes, as shown in the Figure 7, with Newman algorithm having the best and PAM the worst speedup in our tests.In this paper, by using the Newman algorithm for parallel study, and the algorithm compared with the other three algorithms. The test results show that the parallel algorithm of error rate is smaller, and it has very good validity. Especially, the dataset size is bigger, the efficiency will be higher.

REFERENCES

[1] C. Bohm, S. Berchtold and H. P. Kriegel, "Mul-tidimensional index structures in relational databases", Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DaWak 99), Florence, Italy, F, (1999) August 30-September 01.

[2] J. Dean and S. Ghemawat, "Usenix. MapReduce: Sim-plified data processing on large clusters", Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI 04), San Francisco, CA, F, 2004 December 06-08.

[3] L. Bo-hu, C. Xu-dong and H. Bao-cun, "Networked Modeling & Simulation Platform Based on Concept of Cloud Computing—Cloud Simulation Platform", Journal of System Simulation (S1004-731X), vol. 21, no. 17, (2009), pp. 5292-5299.

[4] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", Second Edition [M]. second ed. San Francisco, USA: Morgan Kaufmann, (2006).

[5] H. Xiang, K. Feng-ju and T. Xue-wei, "Research on Framework of Private Cloud in Visual Simulation", Journal of System Simulation (S1004-731X), vol. 23, no. 08, (2011), pp. 1652-1656.

[6] H. An-xiang, F. Xiao-wen and L. Jin-song, "Aviation Simulation Architecture Based on Cloud Computing Platform", Journal of System Simulation (S1004-731X), vol. S1, (2011), pp. 106-109.

[7] R. M. C. Mccreadie, C. Macdonald and I. Ounis, "On Single-Pass Indexing with MapReduce", New York, USA: Assoc Computing Machinery, (2009).

[8] R. Lammel, "Google's MapReduce programming model – Revisited", Science of Computer Programming (S0167-6423), vol. 70, no. 1, (2008), pp. 1-30.

[9] C. Moretti, K. Steinhaeuser and D. Thain, "Scaling Up Classifiers to Cloud Computers", Proceedings of the IEEE International Conference on Data Mining, Pisa, Italy, F, USA: IEEE Computer Society, (2008).

[10] D. Gillick, A. Faria and J. Denero, "MapReduce: Dis-tributed Computing for Machine Learning", [2011-07]. http://www.icsi.berkeley.edu/~arlo/publications/gillick _cs262a _proj.pdf, (2006).

[11] L. Yang and Z. Shi, "An Efficient Data Mining Framework on Hadoop using Java Persistence API", The 10th IEEE International Conference on Computer and In-formation Technology (CIT-2010). Bradford, UK. USA: IEEE, (2010).

[12] S. Hinz, P. Dubois and J. Stephens, MySQL Cluster [M/OL] [08-02-2009] http:// dev.mysql.com/ doc/ refman/ 5.0/ en/ mysql-cluster -overview.html, (2009).

[13] R. Biswas and E. Ort, "Java Persistence API - A Simpler Programming Model for Entity Persistence", http:// java.sun.com/ developer /technicalArticles/ J2EE/jpa/, (2009).

[14] X. Lu, L. Zha and Z. Xu, "Asset-Leasing Model, Architecture, and Key Technology of Vega LingCloud", Journal of Computer Research and Development (S1000-1239), (2010).

[15] C. Bunch, B. Drawert and M. Norman, "Mapscale: a cloud environment for scientific computing", Technical Report, University of California, Computer Science Department, (2009).

[16] L. Kaufman and P. Rousseeuw, "Finding Groups in Data an Introduction to Cluster Analysis", Wiley Interscience, New York, (1990).

[17] C. Pomerance, "A tale of two sieves", Notices of the American Mathematical Society, vol. 43, (1996), pp. 1473-1485.

[18] R. Pike, S. Dorward, R. Griesemer and S. Quinlan, "Interpreting the data: parallel analysis with Sawzall", Scientific Programming, vol. 13, (2005), pp. 277-298.

[19] K. van der Raadt, Y. Yang and H. Casanova, "Practical divisible load scheduling on grid platforms with APST-DV", Proc. of the 19th IPDPS'05, (2005), pp. 29.b.

[20] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski and C. Kozyrakis, "Evaluating MapReduce for multi-core and multiprocessor systems", Proceedings ofInternational Symposium on High Performance Computer Architecture, HPCA, (2007), pp. 13-24.