International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 6 (2015) pp. 15407-15415

http://www.ripublication.com

New Approach For Big Data Mining Using Mapreduce Techniques

Satish Londhe[a*],Smita Mahajan[b,]
[a]*M.Tech student, Symbiosis International University,,Pune, 412115, India*
[b]*Research Guide, Symbiosis International University, Pune, 412115, India*
Email: Satish..Londhe@sitpune.edu.in
Email:smita.Mahajan@sitpune.edu.in

Abstract

The goal of data mining is to discover hidden useful information in large databases. Mining frequent patterns from transaction databases is an important problem in data mining. As the database size increases, the computation time and required memory also increase. Base on this, we use the MapReduce programming mode which has parallel processing ability to analysis the large-scale network. All the experiments were taken under Hadoop, deployed on a cluster which consists of commodity servers. Through empirical evaluations in various simulation conditions, the proposed algorithms are shown to deliver excellent performance with respect to scalability and execution time.

Keywords: Parallel algorithm, Map Reduce, Hadoop, data mining.

Introduction

In Big data the information comes from multiple, heterogeneous, autonomous sources with complex relationship and continuously growing. up to 2.5 quintillion bytes of data are created daily and 90 percent data in the world today were produced within past two years [1].for example Flicker, a public picture sharing site, where in an average 1.8 million photos per day are receive from February to march 2012[10].this shows that it is very difficult for big data applications to manage, process and retrieve data from large volume of data using existing software tools. It's become challenge to extract knowledgeable information for future use [15].There are different challenges of Datamining with Big Data. We overlook it in next section. Currently Big Data processing depends upon parallel programming models like MapReduce, as well as providing computing platform of Big Data services. Data mining algorithms need to scan through the training data for obtaining the statistics for solving or optimizing model parameter. Due to the large size of data it is becoming expensive to analysis data cube. The Map-Reduce based approach is used for data cube materialization and

mining over massive datasets using holistic (non algebraic) measures like TOP-k for the top-k most frequent queries. MR-Cube approach is used for efficient cube computation.

Related Work

Business intelligence and data warehouse can handle TB level data or even higher level. Although many methods have been proposed to deal with high-dimensional data, but the query process is a bottleneck [1]. The emergence of cloud computing to the massive data mining, Hadoop is a MapReduce programming model and mass data [2]. It has made a lot of simulation system in the cloud computing, such a calculation based on the concept of cloud modeling and simulation platform of COSIM-CSP system [3], a new mode of the networked manufacturing [4], private cloud framework for visual simulation [5] ,and the military training system [6]. A simple MapReduce index is completed by McCredie on the Hadoop [7].Ralf proposed a basic program designed to support cloud computing [8]. Moretti introduced a scalable data mining method, the data and computation is distributed to a cloud computing [9]. Gillick implementation of the inquiry learning with Hadoop[10].Many algorithms are using the tree structure in these application systems, such as ID3, C4.5, Fpgrowth KD-tree, PrefixSpan and BIRCH. Most data mining algorithms are based on object oriented programming (OOP) which are usually run on a single computer. Some literatures have been described the detail of method [11-14].However, the characteristics of the MapReduce mode is not suitable for data mining. First of all, MapReduce is lack of overall. The lack of data sharing between the tasks nodes in Hadoop, such as shared memory. The KD tree model and cluster tree model is a model of data mining that requirement for obtaining the global access from the training data. Each node of the Hadoop only processes the data block which is allocated, and output the results of data block. No correlation between the Map sub tasks and between sub tasks of Reduce also unrelated. Therefore, to complete the task of global way linked list and tree structure is very difficult. Secondly, HDFS does not allow random write operation. Because the Hadoop object is GB, or even TB level data. Each file in a block unit distribution is stored in different nodes. It will cause the access failure of subsequent block when we insert or update the data in the middle of the file, so the random write operation is not allowed. Massive data once written into the HDFS (Hadoop distributed file system) will not modify, only can be added or deleted. This ensures that large files can be distributed in each node, and the node only processes the most easily accessible file fragments. Therefore, it is impossible to simulate the list and tree structure by using the HDFS file system. Thirdly, the task has a short life cycle. Each data block is node scanning one time; it will no longer be accessed. In data mining, often training model for data set, and the test set using the model to calculate the effectiveness. As mentioned before, there is no public area to accumulate the results of data analysis in Hadoop, so it is unable to realize the reference in front of the data mining results. Finally, the MapReduce has also been shown to have significant problems [15] with more complex algorithms, like conjugate gradient, fast Fourier transform and block tri-diagonal linear system solver. Moreover, most of

these problems use iterative methods to solve them, indicating that MapReduce may not be well suited for algorithms that have an iterative nature. However, there is more than one type of iterative algorithm. To study if MapReduce model is unsuitable for all iterative algorithms or only a certain subset of them, we devised a set of classes for scientific algorithms. Algorithms are divided between these classes by how difficult it is to adapt them to the MapReduce model and their resulting structure. To be able to compare the classes to each other, we selected and adapted algorithms from each class to the MapReduce model and studied their efficiency and scalability. Such a classification allows us to precisely judge which algorithms are more easily adaptable to the MapReduce model and what kind of effect belonging to a specific class has on the parallel efficiency and scalability of the adapted algorithms. In order to solve the shortcomings and the problems, this paper uses a parallel search algorithm based on MapReduce, to improve the processing speed of massive data. The test proves that the parallel algorithm is scalable to large data sets and finally data will display on secrecy view to end user for more secure approach.

Benjamin C. M. Fung, Ke Wang, Philip S. Yu [21]:
In this Paper the top down specialization algorithm is implemented for the generalization which is enforced by specializing or particularisation the amount of data in an exceedingly top-down manner until a minimum privacy demand is profaned. For handling the categorical and continuous attributes the top-down specialization approach is the feasible way. By minimizing the privacy specification and maximising the data utilization the top down approach uses iterative method to convert the general information into special information. Multiple anonymity issues can be handled with this approach.

Xuyun Zhang, Laurence T. Yang, Senior Member, IEEE, Chang Liu, and Jinjun Chen [22]:
In this paper, they used TDS to measure the drawback of large-scale data anonymization, and introduced a new method called Two-Phase TDS approach which uses Map and Reduce phase on cloud. In this approach first data partition is done and the anonymization is done in parallel as initial state then further intermediate results are produced. In the second section the intermediate results are incorporated with additional anonymization for providing k-anonymous data sets which are persistent. Data anonymization is creatively applied on cloud using MapReduce and it is implemented in the way to produce a highly climbable specialization result. In the experimental results, the scalability and efficiency of large scale data sets has been improved in the Two Phase TDS compared to the centralized TDS.

Ke Wang, Philip S. Yu, SouravChakraborty [23]:
To discover helpful patterns, they explained another optimistic use of the data mining technology even if we mask private information. The bottom-up generalization converts the specific data to less specific but semantically consistent data for privacy preservation and also they focused on two main problems, scalability and quality. The scalability problem was addressed by a unique data structure to focus on pretty good

generalizations. The same quality is achieved by the proposed system however far better measurability compared to existing solutions. Our current algorithm has the likelihood of obtaining stuck at a neighbourhood optimum by greedily hill climbs to a k-anonymity state.

Tiancheng Li, Ninghui Li [24]:
For locating best anonymization, they presented a strategy called bottom-up search. Once the worth of k is small this Strategy works significantly well. They showed the practicability through experiments on real census data for this approach. To find the optimal solution for small k values very quickly, the bottom up approach works efficiently and when k increases, the running time of a generalization scheme increases.

Proposed Work

1. **Data Cube:** Data cube provide multi-dimensional views in data warehousing. If n dimensions given in relation then there are 2^n cuboids and this cuboids need to computed in the cube materialization using algorithm[6]which is able to facilitate feature in MapReduce for efficient cube computation. In data cube Dimension and attributes are the set of attributes that user want to analyse. Cube lattice is formed representing all possible groupings of this attributes, based on those attributes. After that by grouping attribute into hierarchies and eliminating invalid cube regions from lattice we get more compact hierarchical cube lattice. Finally cube computation task is to compute given measure for all valid cube groups. There are different techniques of cube computations [20] like multi- dimensional aggregate computation, BUC (Bottom-Up Computation), star cubing for efficient cube computation. There are limitations in these techniques. There is need of technique to compute cube in parallel on holistic measure over massive dataset. Hadoop based MapReduce can handle large amount of data in cluster with thousands of machines. So this technique is good option for analysis of data.
2. **Map Reduce:** Map Reduce is a programming model designed for processing large volumes of data in parallel by dividing the work into a set of independent tasks. The nature of this programming model and how it can be used to write programs which run in the Hadoop environment is explain by this model. Hadoop [21] is an open source implementation for this environment. Map and Reduce are two functions. The main job of these two functions are sorting and filtering input data. During Map phase data is distributed to mapper machines and by parallel processing the subset it produces <key ,value>pairs for each record. Next shuffle phase is used for repartitioning and sorting that pair within each partition. So the value corresponding same key grouped into {v1, v2…} values. Last during Reduce phase reducer machine process subset <key, {v1, v2,}>pairs parallel in the final result is written to distributed file system. MR1 is used in Hadoop1.0 but due to some resource management issues like inflexible slot configuration, scalability. After Hadoop version 0.23,

MapReduce changed significantly. Now it known as MapReduce 2.0 or YARN (Yet another Resource Negotiator). Map Reduce 2.0 has two major functionalities of job tracker which are spit into resource management and job scheduling into separate daemons [15]. In Hadoop1.0 Job Tracker has a responsibility for managing the resources and scheduling jobs across the cluster. But in Hadoop2.0 the architecture of YARN allows the new Resource Manager to manage the usage of resources across all applications. And Application Masters takes the responsibility of managing the job execution. This new approach improves the ability to scale up the Hadoop clusters to a much larger configuration than it was previously possible. In addition to this, YARN permits parallel execution of a range of programming models. This includes graph processing, iterative processing, machine learning, and general cluster computing.

3. **MR-cube Approach:** MR-Cube MR-Cube is a MapReduce based algorithm introduces for efficient cube computation [7] and for identifying cube sets/groups on holistic measures. MR-Cube algorithm is used for cube materialization and identifying interesting cube groups. Complexity of the cubing task is depending upon two aspects: size of data and size of cube lattice. Size of data impacts size of large group and intermediate size of data, whereas the cube lattice size impacts on intermediate data size and it is controlled by the number/depth of dimension. First we identify the subset of holistic measures that can easily compute in parallel than an arbitrary holistic measure. We can call it Partially Algebraic Measures. The technique of partitioning large groups based on algebraic attribute called Value partitioning. Value partitioning is used to effectively distribute the data; we can easily compute it with Naïve algorithm [5]. Value partitioning performs on only on group that are likely reducer friendly and dynamically adjust the partition factor. Partition factor is ratio by which a group is partitioned. There are different approaches for detecting reducer unfriendly groups. One of the approach is sampling approach where we estimate the reducer unfriendliness of cube region based on the number of groups it is estimated and perform partitioning for all small groups within the list of cube region that are estimated to be reducer unfriendly.
4. **Cube Materialization:** Cube materialization task comes under the MR-Cube approach. Materializing the cube means computing measures for all cube groups satisfying the pruning condition. After materializing cube we can identify the interesting cube groups for cube mining algorithm. The main MR-CUBE-MAP-REDUCE task is perform using annotated lattice. The combine process of identifying and value partitioning unfriendly regions followed by partitioning of regions is referred as annotate. Based on the sampling results cube regions have deemed as reducer unfriendly and require partitioning. Each tuple in dataset the MR-Cube-Map emits key: value pairs for each batch area. In required keys are appended with hash based on value partitioning. The shuffle phase then sorts them by key yielding reducer tasks. The BUC algorithm is then run on each reducer and cube aggregates are generated. The

value partitioned group are merged during post processing to produce the final result.

Secrecy View

Given a micro data table T, a slicing of T is given by an attribute partition and a tuple partition. For example, suppose tables a and b are two sliced tables. In Table a, the attribute partition is {{Age}, {Gender}, {Zipcode}, {Disease}} and the tuple partition is {{t1; t2; t3; t4}, {t5; t6; t7; t8}}. In Table b, the attribute partition is {{Age, Gender}, {Zipcode, Disease}} and the tuple partition is {{t1; t2; t3; t4}, {t5; t6; t7; t8}}.

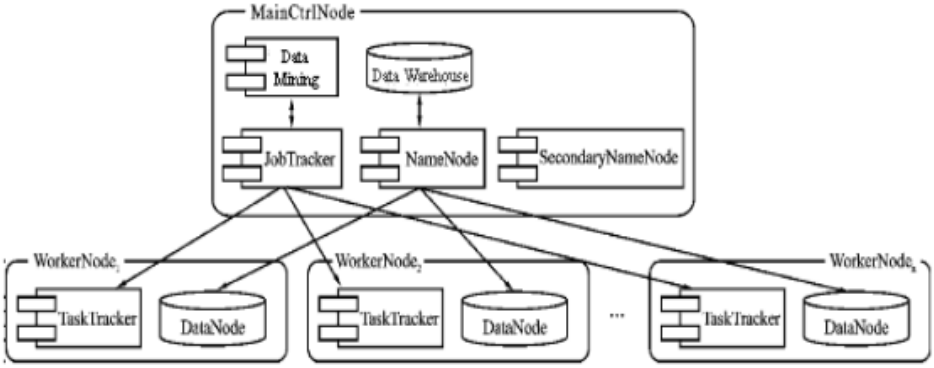

Figure 1: Parallel Data Processing on Hadoop Using Map Reduce

Results

We have done setting up the Hadoop 2.0 cluster nodes. Client-Server communication between Hadoop clients and servers (e.g., the HDFS client to NameNode protocol, or the YARN client to Resource Manager Protocol). System first generates a high dimension health care data and load on database with end user GUI. The different SQL queries show the execution time as well algorithm complexity. The below table shows the algorithm performance.

Table 1: Client Query Execution Performance

Instances	Data Nodes	Times in Seconds
100000	2	**45** seconds
200000	2	**82** seconds
300000	2	**121** seconds
500000	2	**196** seconds

Table 2: Data Secrecy View Form

Name	Address	Age	Zip Code	Disease
****	London	[25-50]	426***	cancer
****	Tokyo	[51-75]	903***	cough
****	Mumbai	[0-25]	399***	tumor
****	Chennai	[0-25]	658***	flue

Future Scope

The system we can implement with YARN database on cloud base architecture, the Software as Service (SaaS), Product as Service (PaaS), Infrastructure as Service (IaaS) will provide the better quality result if working with HDFS.

Conclusion

The MapReduce framework is one of the most imperative parts of big data dispensation. In earlier kinds of MapReduce the mechanisms were designed to address basic needs of processing and resource management. More recently, it has progressed into a much improved version known as MapReduce 2/YARN that provides improved features andfunctionality with HIVE.

In this research work we focused on performance modelling and prediction of Hadoop Map-Reduce systems, the most popular framework for large-scale data processing. We developed the capability to evaluate application performance in hypothetical MapReduce systems using simulation. Compared to the traditional build-and-measure approach, our simulation-based evaluation is faster and cheaper and offers flexibility. Although real experiments must be conducted before total commitment, simulation-based evaluation can work as a intermediate step to reveal obvious flaws and help system designers further understand performance characteristics of their applications and the MapReduce system.

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. Map Reduce: Sim-plified data processing on large clusters", Proceedings of the 6 th 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 /technical Articles/ J2EE/jpa/, (2009).

[14] X. Lu, L. Zha and Z. Xu, "Asset-Leasing Model, Architecture, and Key Technology of Vega Ling Cloud", 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 Inter science, 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 Map Reduce for multicore and multiprocessor systems", Proceedings of International Symposium on High Performance Computer Architecture, HPCA, (2007), pp. 13-24.

[21] Benjamin C. M. Fung, Ke Wang, Philip S. Yu, "Top-Down Specialization for Information and Privacy Preservation", Proceedings of the 21st International Conference on Data Engineering (ICDE 2005), 1084-4627/05 $20.00 © 2005 IEEE.

[22] Xuyun Zhang, Laurence T. Yang, Senior Member, IEEE, Chang Liu, and Jinjun Chen, Member, IEEE, "A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using Map Reduce on Cloud", IEEE Transactions On Parallel And Distributed Systems, Vol. 25, No. 2, February 2014.

[23] Ke Wang, Philip S. Yu, Sourav Chakraborty, "Bottom-Up Generalization: A Data Mining Solution to Privacy Protection", Fourth IEEE International Conference on Data Mining, 2004.ICDM '04

[24] Tiancheng Li, Ninghui Li, "Optimal k-Anonymity with Flexible Generalization Schemes through Bottom-up Searching", ICDM Workshops 2006. Sixth IEEE International Conference on Data Mining Workshops, 2006.