A Cloud-based Efficient On-line Analytical Processing System with Inverted Data Model

Sheng-Wei Huang[1], Ce-KuenShieh[1], Che-Ching Liao[1], Chui-Ming Chiu[1], Ming-Fong Tsai[2] and Lien-Wu Chen[2]
Institute of Computer and Communication Engineering,
Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan[1]
Department of Information Engineering and Computer Science,
Feng-Chia University, Taichung, Taiwan[2]

swh@ee.ncku.edu.tw, shieh@ee.ncku.edu.tw, liaochei@gmail.com, asserchiu@gmail.com, tsaimf@fcu.edu.tw and lwuchen@fcu.edu.tw

Abstract

On-line analytical processing (OLAP) provides analysis of multi-dimensional data stored in a database and achieves great success in many applications such as sales, marketing, financial data analysis. OLAP operation is a dominant part of data analysis especially when addressing a large amount of data. With the emergence of the MapReduce paradigm and cloud technology, OLAP operation can be processed on big data that resides in scalable, distributed storage. However, current MapReduce implementations of OLAP operation processing have a major performance drawback caused by improper processing procedure. This is crucial when dimension or dependent attributes are large, which is a common case for most data warehouses hold nowadays. To solve this issue, this paper proposes a methodology to accelerate the performance of OLAP operation processing on big data. We have conducted the experiments on the basic algebra of OLAP operation with different data sizes to demonstrate the effectiveness of our system.

1. Introduction

The idea of Big Data is becoming popular today. Not only do we focus on the information produce by each person but the world which is fell with Big Data. Strictly speaking, the definition of Big Data has not been defined so far. Most of the statements believe that the data set is called Big Data which cannot be managed, stored or analyzed by traditional software database management tools and hardware environment. Over the past few years, a large amount of researches have conducted the experiments to show that cloud computing technology is well-suited for processing Big Data. The cloud technology has the advantage of scalability and provides efficient parallel and distributed computing environment. Engineers in Google Company published a paper which is the discussion about the brand new programming model called MapReduce [3]. This framework can process Big Data in cloud computing environment. Moreover, programmer can utilize the resource of the large scale distributed system easily even if they have no programming experience of cloud computing. Apache Hadoop [4] is a successful distributed computing technology that suited for MapReduce programming model. It is designed for the analysis of Big Data and provides the cloud computing environment. The scaling mechanism of thousands of nodes on Hadoop cluster can be achieved with highly availability. Apache HBase [6] is developed as part of Hadoop project. It is an open source, distributed database model after Google's BigTable [7]. The storage layer of HBase is built on top of HDFS [5]. It is a column-oriented storage system with highly scalability. Therefore, we choose Hadoop and HBase as computing environment and MapReduce as programming model.

Data warehouse [8, 9, 13, 16] provides architectures and tools for organization to make strategic decisions. It consolidates data in multidimensional space. OLAP (online analytical processing) [1] is a new technology for interactive analysis of multidimensional data, which facilitates effective mining of data. It is the most important tool for data analysis in data warehouse system. Hence, data warehouse is a platform for data analysis and OLAP is a step in the knowledge discovery process. Moreover, OLAP operations can be

QSHINE 2015, August 19-20, Taipei, Taiwan

DOI 10.4108/eai.19-8-2015.2261409

integrated with data mining functions, such as clustering, prediction and so on. It enhances the interactive mining of knowledge. Data warehouse and OLAP form an essential step in data intensive analysis which can help knowledge discovery in science, business and industry. A given query of OLAP application is transformed into a set of relational algebra [14], such as σ (SELECT) algebra and π (PROJECT) algebra and then is optimized processing. These queries allow retrieval of specified subsets of data [12]. For example, slicing and dicing a data cube may correspond to σ and π algebra respectively.

With the fast development of OLAP applications, information stored in the database presents exponential growth. These data sets are so large that it becomes difficult to process using traditional database management tools or data processing applications. The performance of OLAP is restricted by using the traditional database management system, such as row-oriented storage model [2]. The original idea of OLAP shows an important point of view that the operation throughput and response time are much more important than transaction throughput in OLAP applications. It is difficult to perform OLAP operations on Big Data in traditional row-oriented database management system.

OLAP tools have been deployed on distributed computing environment to analyze data sets. Hadoop provides an appropriate computing environment for OLAP tools. HBase is column-oriented storage model which can be used to store data from different dimension to different files. The processing procedure of OLAP operation is well-suited on MapReduce programming model because OLAP operation is mostly read-only and taking operations on large amounts of historic data. With the help of these cloud computing technologies, it is friendly to perform OLAP applications on distributed computing environment. OLAP is convenient to utilize on cloud environment.

In this paper, we propose a system methodology which is suited for OLAP operation with Big Data processing. It minimizes the occurrence of a large number of memory-swapping while conducting the OLAP application on the computing nodes in Hadoop cluster. To improve the performance of π (PROJECT) algebra, we implement the Hierarchy Inverted Model which reduces the scanning cost of dimension-oriented storage model effectively.

The rest of this paper is organized as follows: Section 2 describes the related work of OLAP operation processing in Cloud. The Hierarchy Inverted Model in our methodology is illustrated in section 3. The experiments and performance evaluations are shown in section 4, and conclude the paper in section 5.

2. Related Works

2.1.Hive

Hive [10] is an Apache open source data warehousing solution built on top of Hadoop. Hive supports queries on OLAP storage model on HBase. It can compile OLAP operations into MapReduce job that executed on Hadoop. Hive also includes a system catalog, Hive-Metastore, containing schemas and statistics. These system components are useful for Hive in data exploration and query optimization. However, we find it hard to achieve comparable performance on the processing procedure of OLAP operation based on Hive. According to our experiments, the Linux memory management system performs a large number of memory-swapping on each computing node of Hadoop cluster while performing OLAP operation on Hive. This condition represents that the executing process occupies a large amount of memory space while conducting the execution job based on Hive. We believe that this is the main reason why it is difficult for Hive to achieve satisfying performance.

2.2. Hadoop OLAP

Hadoop-OLAP is published in 2012. It is an OLAP aggregation system based on Hadoop and HBase. Hadoop-OLAP includes dimension-oriented storage model, efficient data index and hierarchy character of data. Based on these system components, the I/O cost can be reduced significantly while performing a part of OLAP operation. The processing procedure of OLAP operation for Hadoop-OLAP is based on MapReduce programming framework which achieves highly concurrency and parallel computing for aggregation job. It is well suited for multidimensional data since it process dataset form different dimensions simultaneously. However, we find that Hadoop-OLAP is suited for only SELECT algebra of OLAP operation. It is hard to achieve comparable performance while performing PROJECT algebra of OLAP operation. The attribute or tuple reconstruction issue [11] may cause the large amount of data transferring and disk I/O in Hadoop cluster based on Hadoop-OLAP computing system. According to the introduction of Google's MapReduce paper, the unnecessary data transfer and disk I/O in Hadoop cluster will cause bottlenecks severely. We believe that this drawback is the main reason why it is difficult for Hadoop-OLAP to achieve satisfying performance while performing a part of important OLAP operations.

3. System Design and Inverted Data Model

Our methodology contains an OLAP operation processing engine and a storage system. The computing system of processing engine is designed based on MapReduce programming model. We use Hadoop as a framework to integrate the processing engine and MapReduce. In the storage system design, we choose Apache HBase as our data store. Figure 1 presents the system architecture of our system.

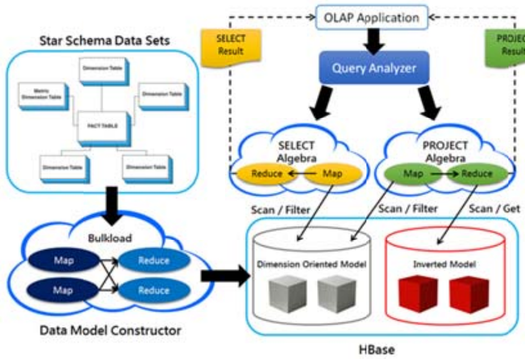

Figure 1. System architecture

The Data Model Constructor is design for creating Dimension-oriented model and Inverted Model with MapReduce. We use MapReduce as the programming model to speed up the processing time of model construction for the system.

We use a well-known benchmark, SSBM (TPC-H), to generate the testing data sets with three kinds of scales factors. We take the Dimension-oriented Model of Hadoop-OLAP as the reference and add up an Inverted Model for our system which is suited for the algorithm of algebra execution based on HBase. According to our experiment, the idea of Dimension-oriented Model with Inverted Model is quite useful for the purpose of speed up the processing time of PROJECT algebra.

The idea of Dimension-oriented Model is based on Value Code and Dimension Hierarchy Code.

- Value Code: The relationship between tuples in an attribute is often in a form of hierarchy and there is a Value Code for each kind of tuples.
- Dimension Hierarchy Code (DHCode): In a dimension tree, the Dimension Hierarchy Code is encoded by the Value Code of non-root node in the form of top-down mode. It represents the hierarchy relationship between tuples. This kind of model is quite useful for range query and aggregation. The Dimension Hierarchy Code is stored in the index table on HBase. The index table of Dimension Hierarchy Code is shown as Figure 2. In dimension-oriented Model, data form different dimension is stored in individual table. Consider the virtual view of region attribute in Figure 3.

Dimension Hierarchy Code	
001-00N-000	Taipei
001-00N-001	Taoyuan
001-00N-00N	Taichung

Figure 2. Index table of DHCode

Fact ID	Region		
	Continent	Country	City
1	Asia	Taiwan	Taipei
2	Asia	Taiwan	Taichung
3	Asia	Taiwan	Taoyuan
4	Asia	Taiwan	Taipei

Figure 3. Virtual view of Region attributes

We store the corresponding Dimension Hierarchy Code in HBase table as shown in Figure 4. The Value List in Column Family is the Primary Key of Region attributes which correspond to the dimension attribute in Fact table.

With the help of dimension-oriented Model, it reduces the unnecessary I/O cost of scanning when performing σ algebra processing. However, there is a room for performance improvement of PROJECT algebra. Therefore, we add the Inverted Model to our system.

Row Key : DHCode	Column Family : Value List
001-00N-000	1, 4
001-00N-001	3
001-00N-00N	2

Figure 4. DHCode in HBASE table

Row Key : Fact	Column Family : DHCode
1	001-00N-000
2	001-00N-00N
3	001-00N-001
4	001-00N-000

Figure 5. Inverted model in HBASE table

The difference between Inverted Model and Dimension-oriented Model is the storing structure of HBase table. The example of Inverted Model table correspond to Dimension-oriented table (Figure 4) is shown in Figure 5.

4. Performance Evaluation

Most of OLAP operations are based on two kinds of algebra, which are π algebra and σ algebra. To focus on the performance comparison between Hadoop-OLAP and our methodology, we show the performance on the π algebra.

4.1. Experimental Setup

We evaluate the performance of our system in a 12-worker cluster, each worker equipped with an Intel Xeon E3-1240 CPU. The processor's clock rate is 3.30GHz and the numbers of core is four. Besides, each worker also has 8 GB main memory and 1.34TB disk space. Those workers connected via Gigabit network. As for the software we installed Hadoop-1.0.3, HBase 0.94.4 and Java 1.7.0 in each node.

To prove our approach is effective in OLAP operations, we test our approaches by using the well-known benchmark TPC-H (SSBM). TPC-H (SSBM) is an acronym of Transaction Processing Performance Council-H (Star Schema [15] Based) and it provides an explanation of design of decisions made in creating the Star Schema benchmark. In order to meet require of the idea of big data volume of Big Data, we use the factors which are SSBM (2500), SSBM (5000) and SSBM (10000) with data size 250GB, 500GB and 1TB respectively. We take the approach Hadoop-OLAP and Hive as our compared system.

4.2. Experimental Results

In the section, we present the evaluation results on the PROJECT algebra. We give the results of two different kinds of selectivity. Wide selectivity represents a bigger slice while the narrow selectivity represents a smaller slice relatively. Since the main idea of OLAP operation processing is its response time, we show the processing time of the PROJECT algebra. We also show another factor memory usage to present a more complete experiment.

First, we present a narrow selectivity of π Algebra (Operation 1) with two different factors.

$$\pi_{<Date>}\left(\sigma_{<REGION=ASIA.China \wedge Total-price=10000000>}\left(Seller\right)\right)$$

Operation 1. Narrow selectivity of PROJECT algebra

The experiment results are shown in Figure 6 and Figure 7. The processing time of our methodology is faster than others. Moreover, our method performs less memory usage comparing to other system.

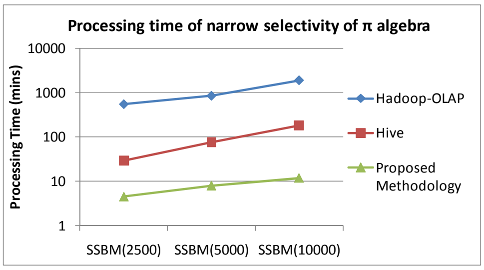

Figure 6. Processing time of narrow selectivity of π algebra

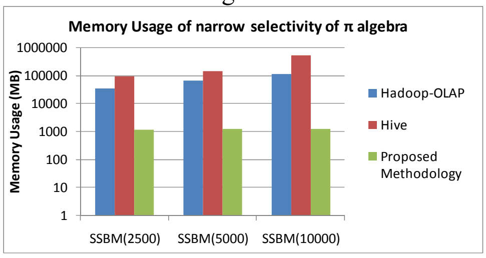

Figure 7. Memory usage of narrow selectivity of π algebra

Second, we present a wide selectivity of π Algebra (Operation 2) with two different factors.

$$\pi_{<Price>}\left(\sigma_{<REGION=ASIA.China \wedge Date=1998.January>}\left(Seller\right)\right)$$

Operation 2. Wide selectivity of PROJECT algebra

The experiment results are shown in Figure 8 and Figure 9. The processing time and memory usage of our methodology still outperforms other systems.

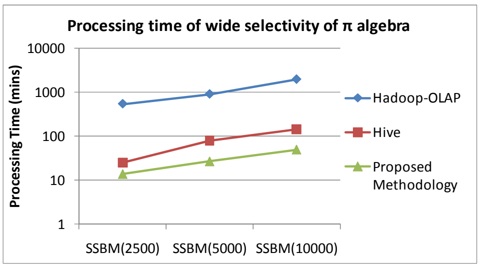

Figure 8. Processing time of wide selectivity of π algebra

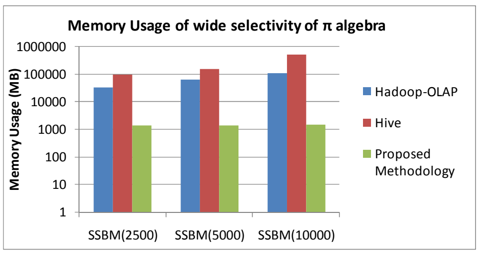

Figure 9. Memory usage of wide selectivity of π algebra

5. Conclusions and Future Work

In our approach, we construct a Dimension Model with Inverted Model on HBase storage system and an OLAP operation processing engine that implement based on MapReduce. In HBase storage, we use two table schemas in order to process both of the basic OLAP algebras efficiently. The system has been verified that it performs better than other OLAP operation processing system in the experimental results.

In our future work, we decide to compress both of the dimension model and inverted model. Since there are large amount datasets stored in data warehouse, using two kinds of models with uncompressed data may be harmful to storage space. Moreover, we will enhance the system to be more robust with more convenient using interface for user. We will keep optimizing our system and hope that it will become a useful tool for OLAP operation processing on data warehouse.

Acknowledgement

This work is supported by the Ministry of Science and Technology of Republic of China(R.O.C) with the project numbered as MOST103-2218-E-035-013, MOST104-2221-E-035-021 and NSC101-2221-E-006-147-MY3.

References

[1] Chaudhuri, Surajit, and UmeshwarDayal, "An overview of data warehousing and OLAP technology," ACM Sigmod record, volume 26, issue 1, pp. 65-74, 1997.

[2] Jing-hua, Zhao, Song Ai-mei, and Song Ai-bo, "OLAP Aggregation Based on Dimension-oriented Storage," in Proceedings of IEEE 26th Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012.

[3] Dean, J. and S. Ghemawat, "MapReduce: simplified data processing on large clusters," Communications of the ACM, volume 51, issue 1, pp. 107-113, 2008.

[4] Hadoop. Available from: http://hadoop.apache.org/

[5] Borthakur, Dhruba, "HDFS architecture guide," HADOOP APACHE PROJECT, http://hadoop.apache.org/common/docs/current/hdfsdesign.pdf

[6] Apache HBase. Available from: http://hbase.apache.org/

[7] Chang, Fay, et al. "Bigtable: A distributed storage system for structured data," ACM Transactions on Computer Systems (TOCS), volume 26, issue 2, pp. 1-14, 2008.

[8] Ramez Elsmari and Sham Navathe, "Fundamentals of database systems, 6th edition," Pearson , 2010.

[9] William H. Inmon, "Building the data warehouse," John wiley& sons, 2005.

[10] Thusoo, Ashish, et al. "Hive-a petabyte scale data warehouse using Hadoop," in Proceedings of Data Engineering (ICDE), 2010 IEEE 26th International Conference on.

[11] He, Yongqiang, et al. "RCFile: A fast and space-efficient data placement structure in MapReduce-based warehouse systems," in Proceedings of Data Engineering (ICDE), 2011 IEEE 27th International Conference on.

[12] Han, Jiawei, and Micheline Kamber. "Data Mining, Southeast Asia Edition: Concepts and Techniques." Morgan Kaufmann, 2006.

[13] Giovinazzo, William A. "Object-oriented data warehouse design: building a star schema," Prentice Hall PTR, 2000.

[14] Davis, Martin, and Hilary Putnam, "A computing procedure for quantification theory," Journal of the ACM (JACM), volume 7, issue 3, pp. 201-215, 1960.

[15] E. B. O. Patrick O'Neil and X. Chen, "The Star Schema Benchmark," http://www.cs.umb.edu/~poneil/StarSchemaB.PDF, 2007

[16] Meenakshi Srivastava, S.K.Singh and S.Q.Abbas, "An Architecture for Creation of Multimedia Data Warehouse," International Journal of Engineering Science and Innovative Technology (IJESIT), Volume 2, Issue 4, pp. 309-315, 2013.