A REVIEW DATA CUBE ANALYSIS METHOD IN BIG DATA ENVIRONMENT

Dewi Puspa Suhana Ghazali[1], Rohaya Latip[1, 2], Masnida Hussin[1] and Mohd Helmy Abd Wahab[3]
[1]Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia (UPM), Serdang, Selangor, Malaysia
[2]Institute of Mathematical Research, Universiti Putra Malaysia (UPM), Malaysia
[3]Faculty of Electrical and Electronic Engineering, Universiti Tun Hussin Onn Malaysia, Batu Pahat, Johor, Malaysia
E-Mail: dewipuspasuhana@gmail.com

ABSTRACT

With the development of computer technologies, the amount of data has explosive growth and the data volumes have approximately doubled each year. One of the tools that affect scalability and flexibility to handle structured as well as unstructured data called Hadoop. However, data cubes are widely used as a powerful tool to provide multi-dimensional views in data warehousing and On-Line Analytical Processing (OLAP). Therefore, it is becoming expensive to perform the data cube analysis when the data sizes increases. In this paper, we will investigate and review the methods of MapReduce in Hadoop.

Keywords: big data, data cube, MapReduce.

INTRODUCTION

Big data has become a new trend where society becomes more instrumented because big data exist in many applications such as web logs, sensor networks, and cloud computing (Hsu I-C, 2013) as a result, organizations are producing and storing vast amounts of data. But, these issues are the most significant issues in the industrial area, such as manufacturing, sales, transportation and finance because the organization needs to make decisions based on aggregation of data over multiple dimensions (D. J. DeWitt and J. Gray, 1992). There is broad recognition of the value of data, and products obtained through analyzing it. A Popular powerful tool for analyzing multidimensional data is Data Cube Analysis (J. Gray; S. Chaudhuri; A. Bosworth; A. Layman; D. Reichart; M. Venkatrao; F. Pellow, and H. Pirahesh, 1997). Data Cube Analysis is a powerful tool for analyzing multidimensional data.

The Data Cubes (Li, Z., Yang, C., Jin, B., and Zhan, M *et al*., 2015) are one such critical technology that has been widely used in data warehousing and On-Line Analytical Processing (OLAP) for data analysis in support of decision making. In OLAP, the attributes are classified into 1) dimensions (the grouping attributes) and 2) measures (the attributes which are aggregated) (Jeffrey Dean *et al*, (2004). The Data Cube is used to represent data along some measure of interest and one of the easy way to look at the data that allow us to look at complex data in a simple format. Queries are performed on the cube to retrieve decision support information. Called a “cube” because it can be 2- dimensional, 3- dimensional, or higher-dimensional. Cube designing is efficient for handling of unstructured data.

There are two techniques used in cube analysis: First, designed for a small number of nodes means for a single machine or cluster. Second, many of the establishes techniques take advantage of the measure being algebric (J. Gray; S. Chaudhuri; A. Bosworth; A. Layman; D. Reichart; M. Venkatrao; F. Pellow and H. Pirahesh, 1997). and use this property to avoid processing groups with a large number of tuples.

The Data Cube operator must calculate all subsets of the dimensions specified in the operation. The cube operation can be expressed in high-level MapReduce languages (Wang, Z.; Chu, Y.; Tan, K. L.; Agrawal, D.; Abbadi, A. E., and Xu, X., 2013).

This review journal aims to review the existing approaches in Scalable Data Cube Analysis in Big Data environment and suggest potential directions in the area. The first part of the paper introduces the MapReduce that uses in the Big Data environment. Next, explains and summarizes methods on the related literature review such as journal and papers.

RELATED WORK

Introduction of data warehouse and OLAP cube

In this section, we briefly explain what is Data warehouse and OLAP Cube. Before intro more about Mapreduce, we will explain about the Cube types and their concepts first in this section too.

A data warehouse is the main repository of the organization's historical data, its operate memory (Mumick, I. S.; Quass, D. and Mumick, B. S., 1997). A data warehouse is a relational database that is designed for query and analysis. It usually contains historical data driven from transaction data, but it can include data from other sources. More, formally, Bill Inmon (Mumick, I. S.; Quass, D. and Mumick, B. S., 1997) is one of the earliest and most influential practitioners where defined a data warehouse as follows:

1) Subject oriented
2) Time variant
3) Non-volatile
4) Integrated
5) Offline operational databases
6) Offline data warehouse
7) Real time data warehouse
8) Integrated data warehouse

A data warehousing provides architecture and tools for business executives to systematically organize, understand, and use their data to make strategic decisions. Data warehouse systems serve users or knowledge workers in the role of data analysis and decision making. The system call OLAP that can organize and present data in various formats in order to accommodate the diverse needs of the different users.

OLAP as a computer processing that enables a user to easily and selectively extract and view data from different point of view. OLAP also allows users to analyze database information from multiple database systems at one time and OLAP data are stored in multidimensional database. Then popularly known as a user-friendly environment for interactive data analysis. Example some of the popular OLAP server software programs: 1) Oracle Express Server 2) Hyperion Solution Essbase. Basically OLAP products are designed for multiple-user environment, with the cost of the software based on the number of users. OLAP functions is very efficient in queries because OLAP can determine which operation should be performed on the available cuboids also can determine to which materialized cuboid(s) the relevant operations should be applied. Table-1. shows summarize differentiate between a data warehouse and OLAP.

Table-1. Differentiate a data warehouse and OLAP.

Function	Data warehouse	OLAP
Operation	Report	Analyze
Analytical requirements	Medium	High
Data level	Medium and summary	Summary and derived
Age of data	Historical and current	Historical, current and projected
Business events	Anticipate	Predict
Business objective	Efficiency and adaptation	Effectiveness and design

Data Cube concept

The data cube is the N-dimensional generalization of simple aggregate functions, which is introduced by Gary (J. Gray; S. Chaudhuri; A. Bosworth; A. Layman; D. Reichart; M. Venkatrao; F. Pellow and H. Pirahesh, 1997). In OLAP system, a data cube is a way of organizing data in N-dimensions so as to perform analysis over some measure of interest. The basic cube problem is to compute all the aggregates as efficiently as possible but, for the main difficulty is that the cube in exponential in the number of dimensions. In additional, the size of each group-by depends upon the cardinality of its dimensions. As many techniques are proposed for efficient cube computation. A data cube has four different kinds of cubes which are: 1) Full Cube, 2) Iceberg Cube, 3) Closed Cube, and 4) Shell Cube. Table-1 shows the types of cube and description.

Table-1. Cube types.

Cubes Type	Description
Full Cube	• To ensure fast on-line analytical processing (OLAP), it is sometimes desirable to precompute the full cube.
Iceberg Cube	• To save the processing time and disk space and for more focused analysis. • The cells cannot pass the threshold are likely to be too trivial to warrant further analysis.
Closed Cube	• Data Cube consisting for only closed cells.
Shell Cube	• Precompute cuboids that involve only for a small number of dimensions.

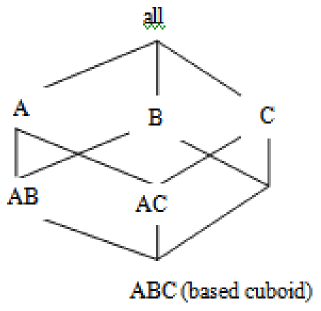

Figure-1. Lattice of cuboids.

Figure-1 shows a 3-D data cube for the dimension A, B and C. ABC is the base cuboid, containing all three of the dimension. A cell in the based cuboids is a base cell. A cell from a non-base cuboids is an aggregate cell. An aggregate cell over one or more dimensions. Next, we will discuss more about method that are using to analyze Data Cube in Big Data Environment.

Data Cube analysis method over Big Data environment

A. MapReduce

MapReduce framework is introduced as a simple and powerful programming model that enables easy development of scalable parallel application to process vast amounts of data on large clusters of commodity machines (Dean, J.Ghemawat, 2004). MapReduce accelerates the processing of large amounts of data in a cloud; thus, MapReduce, is the preferred computational model of cloud providers.

MapReduce was developed by Google for their large data analysis, especially for scanning trillions of web pages to compute the most relevant pages for a search query. The purpose of MapReduce is it can intelligently distribute computation across a cluster with hundreds or thousands of computers, each analyzing a portion of the datasets store locally on the compute node. MapReduce has provided researchers a powerful tool for tackling large-data problem in areas of machine learning, text processing, and Bioinformatics.

Hadoop is an open source implementation of the MapReduce framework running on top of a distributed file system called Hadoop Distributed File System (HDFS). In HDFS, a file is divided into multiple blocks, called splits, which are then replicated on multiple machines on the cluster. The size of a block is configurable. HDFS helps in replication of files when software and hardware failure occurs, and automatically re-replicates the data blocks by providing security of Big Data. **Error! Reference source not found.** shows the architecture of MapReduce.

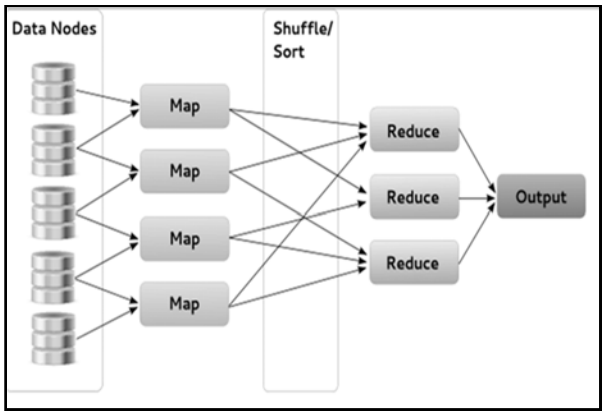

Figure-2. Architecture of MapReduce.

The user of the MapReduce framework expresses the computation using two functions: *Map* and *Reduce.* The Map functions takes an input pair and produces a set of intermediate key, *I* / value, *V* pairs and the architecture will groups all immediate values together associated with the same intermediate *I* and pass to the Reduce function. The Reduce function take an action to receive an intermediate I with its set of values, *V* and merges them together. However, this model does not directly support processing multiple related heterogeneous datasets. While processing relational data is a common need, this limitation causes difficulties and inefficiency when MapReduce is applied on relational operations like joins.

MapReduce advantages

The programmer can take more advantage to deal with irregular or unstructured data because it more easily than they do with DBMS. There are several advantages:

1) As mentioned before, MapReduce is a simple and efficient for computing aggregate. Because with MapReduce, a programmer defines his job with only Map and Reduce functions, without having a specify physical distribution of his job across nodes. MapReduce also flexible because does not have any dependency on data model and schema.

2) MapReduce also popular with the word highly fault tolerance. Example, it is reported that MapReduce an continue to work in spite of an average of 1 or 2 failures per analysis job at Google (Jeffrey Dean et al, 2004).

3) The best advantage of using MapReduce is high scalability. Yahoo! Reported that their Hadoop gear could scale out more than 4,000 nodes in 2008 (A. Anand, 2008).

4) MapReduce is basically independent from underlying storage layers. Thus, MapReduce can work with different storage layers.

MapReduce challenges

When introduced the MapReduce, this method shows that many problems can be solved at scale unprecedented before (Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U., 2015). However, such methods would be necessary to achieve high scalability and fault tolerance in massive data processing. Thus, how to increase efficiency guarantee the same level of scalability and fault tolerance is a major challenge.

MapReduce based approach for Cube computation over Big Data

MapReduce is also used to handle two major issues such as data distribution by illustrating a framework to partition high multidimensional lattice into region areas and distribution of data analysis and mining under parallel computing infrastructure. **Error! Reference source not found.** shows a flow chart for cube computation by using the MapReduce approach.

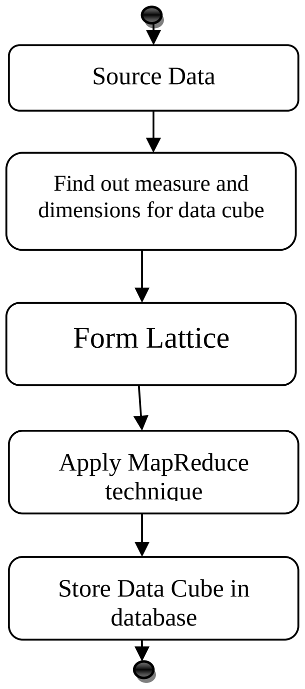
Source Data
Find out measure and dimensions for data cube
Form Lattice
Apply MapReduce technique
Store Data Cube in database

Figure-3. Flow chart for cube computation using MapReduce.

B. MR-Cube

As mentioned, the Data Cube can be expressed in high-level MapReduce (Dean, J. Ghemawat., 2004). Therefore, a study is needed to identify the existing challenges which are: a) Size of Intermideate Data b) Size of Large Groups. The complexity of the cubing task depends on two aspects: a) data size and b) cube lattice size. MR-Cube enhance from MapReduce that can give more efficient in data cube computation.

Data size impacts both intermediate data size and the size of large groups. The cube lattice size impacts the intermediate data size and is controlled by the number/ depth of dimension.

Based on the example of an MR - Cube architecture that is shown in **Error! Reference source not found.**, a user specifies dimension hierarchies and a measure. Then, construct an annotated cube lattice by using a data sample. The process involve in this phase are map, combine, shuffle, and reduce. Then will send to cube materialization Mapreduce phase, in cube materialization, tuples are mapped to each batch area. Reducers evaluate the measure for each batch area. Next, partitioned measure is merged in a post-processing step. The cube is loaded into a Database (P.Pramod; K. Prini; G. Aishwarya *et al*, 2015).

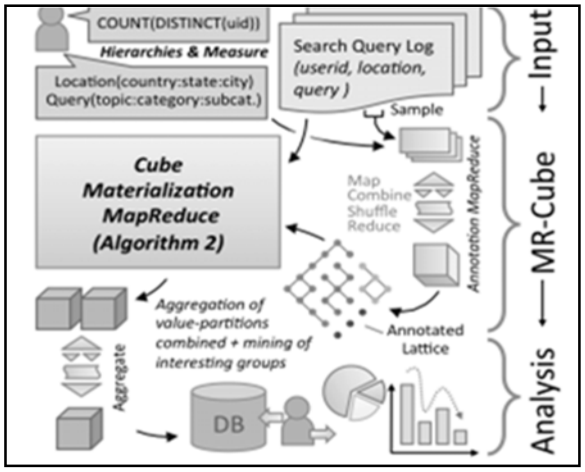
COUNT(DISTINCT(uid))
Hierarchies & Measure
Location(country:state:city)
Query(topic:category:subcat.)
Search Query Log (userid, location, query)
Sample
Cube Materialization MapReduce (Algorithm 2)
Map
Combine
Shuffle
Reduce
Annotation MapReduce
Aggregation of value-partitions combined + mining of interesting groups
Annotated Lattice
Aggregate
DB
Input
MR-Cube
Analysis

Figure-4. MR-Cube architecture.

This MR-Cube are efficiently distributes the computation workload across the machines and is able to complete cubing tasks at a scale where prior algorithms fail. Then two techniques needed for effectively distributed the data and computation workload. 1) Value partitioning and 2) Batch area. Value Partitioning is used for effectively distributed data for that we are going to run Naive algorithm Nandi, A., Yu, C., Bohannon, P., and Ramakrishnan, R., 2012). We want to perform value partitioning only on groups that are likely to be reducer unfriendly and dynamically adjust the partition factor. Therefore, for implementation standpoint, it is advisable to provide both Naive and MR-Cube to the user because while the data is small or the measure is algebraic, the Naive algorithm is recommended. Based on existing work (Nandi, A.; Yu, C., Bohannon; P., and Ramakrishnan; R., 2012). MR-Cube is feasible to perform both large scale cube materialization and mining in the same distributed framework. However, both of this work do not provide a generic algorithm that can balance the load to materialize the cube for different measures. Mr-Cube just provide a solution to a special case 1during the cube computation under MR where one reducer gets the hot spot group-by cell with a large number of tuples.

C. Tile cube in SOLAP

SOLAP or call Spatial On-Line Analytical Processing is a powerful decision support systems tool for exploring the multidimensional perspective of spatial data. The typical Spatial OLAP (SOLAP) cube can be generally described as a tuple. In recent years, remotely sensed data have been integrated into SOLAP cubes, and this improvement has advantages in spatio-temporal analysis for environment monitoring. However, the performance of aggregation in SOLAP still faces a considerable challenge from the large-scale dataset generated by Earth observation. Because of the storage limitation, it is not practical to precompute all the cube measures. Therefore, some of the high-level measures would be computed through online queries and output visually by charts/graph (Li, J., Meng, L.; Wang, F. Z.; Zhang, W., and Cai, Y., 2014). The challenges come out with a new solution that calls Tile Cube.

The implementation of MR enabled Tile Cube includes cube data storage and aggregation in the MR pipeline. A recent paper use HBase to distributed non-relational database that hosts very large tables (made of billions of columns/row) is employed to store cube data. The aggregation of the cube data are implemented on top of Hadoop. The

shows the Tile Cube working experience.

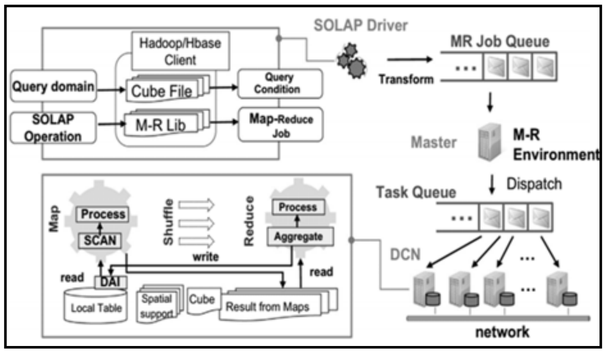

Figure-5. Tile Cube architecture.

described the tile cube architecture. First, the SOLAP driver transforms SOLAP operations to M-R Jobs. At the same time, the data query domain is phrased to query the condition, and the input parameters of M-R Job are configured automatically. After that, the job is submitted via the Hadoop/HBase client to scheduling queue on the master node. When the job gets ready to run, the task workers on DCNs apply subtasks to master for executing the Map/Reduce phase. Then, the SCAN runs distributed data scanning on DCNs via the Data Access Interface (DAI), and each result are consumed as a local input of the Map phase. Finally, the Reduce phase collects the grouped results from Maps, executes the aggregation and writes the finally results into a database (Li, J., Meng; L., Wang, F. Z.; Zhang, W., & Cai, Y., 2014). There are several benefits by implementing this method:

1) Storage = The fact query develops to map the joined cell list and other constraints in the domain to query filters, and executes distributed scanning to find matched tiles.

2) Aggregation = can be decomposed into multiple traditional map algebra function based on the granularity tile.

To make full use of MR in SOLAP, several influencing factors, such as network traffic patterns and computational complexity, should be considered in a balanced way for the specific aggregation. The existing method have some limitation and need future work to support more SOLAP operations and simple data mining more efficient.

D. HaoLap

HaoLap is the enhancement of OLAP by improving the performance and system scalability. HaoLap adopts share-nothing architecture. When OLAP query is processing, HaoLap not only reduces the queried scope but also ensures the parallelism, and distributed OLAP job to data to reduce the data transfer time (Song, J., Guo, C., Wang, Z., Zhang, Y., Yu, G., and Pierson, J. M., 2015). HaoLap is built on Hadoop and based on the proposed models and algorithm:

1) specific multidimensional model to map the dimensions and the measures, 2) the dimension coding and traverse algorithm to achieve the roll up operation on hierarchy of dimension values, 3) the chunk model and partition strategy to shared the cube, 4) the chunk selection algorithm to optimize OLAP performance, and 5) the MapReduce based OLAP and data loading algorithm

There are some reasons leading to the advantages. a) HaoLap draws the experiment of MOLAP; therefore the high performance of OLAP in HaoLap is nature. HaoLap takes advantages of the special data model, simplified dimension model that the basis of HaoLap, to keep the OLAP simple and efficient. b) The dimension coding and traversing algorithms maps the dimension values with measures in an efficient calculation method but heavy index approach.

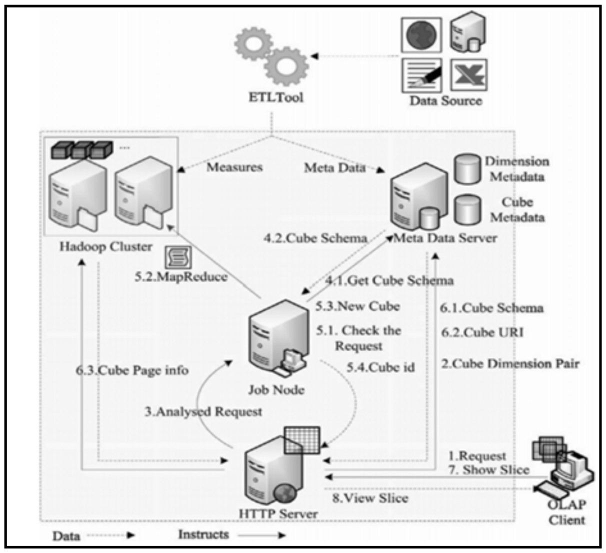

Figure-6 shows the HaoLap system.

VOL. 10, NO 19, OCTOBER, 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2015 Asian Research Publishing Network (ARPN). All rights reserved.

www.arpnjournals.com

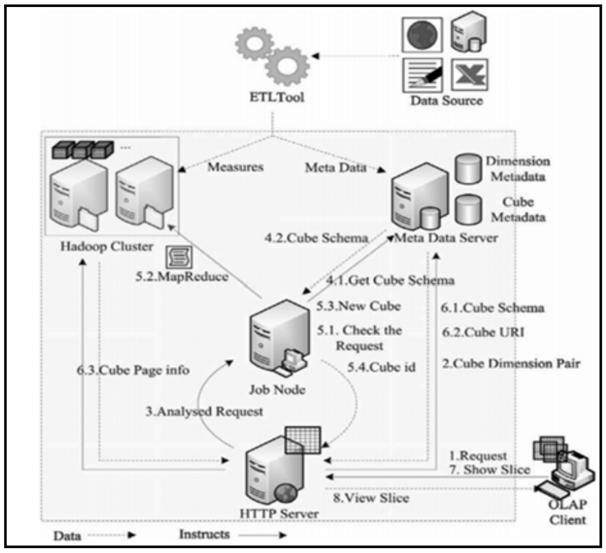

Figure-6. HaoLap system.

HaoLap includes five components which are: Hadoop cluster, Metadata Server, Job Node, OLAP service Facade and OLAP client.

1) Hadoop cluster: The function of Hadoop cluster is to run the job using MapReduce framework if Job node submits a MapReduce job of OLAP. Hadoop cluster also maintains a MapReduce instance and a HDFS instance; therefore measures are stored in Hadoop Cluster.

2) Metadata server: Two main function of Metadata Server. Firstly, the metadata stored in metadata server, which include two parts, such as dimension metadata and cube metadata. The dimension metadata describes the dimension structure, which persisted in XML file. Cube metadata is the cube information. In HaoLap, dimension and cube cost ignorable storage. Secondly, the netadat server is a metadata maintainer receiving instructions from job node.

3) Job node: Job Node is the core of HaoLap that responds for checking processed OLAP commends, generating and monitoring OLAP jobs, and processing metadata with Metadata Server.

4) OLAP service facade (OLAP SF): This section receive original request from OLAP client then generated an intermediate form request and submits the intermediate form to job node and waits for the result. Finally, the OLAP SF reports the result to the OLAP client.

5) OLAP client: The OLAP clients are an application that collects user's input, connect with the OLAP SF, and reports the OLAP SF response to the user. OLAP client also response for showing the cube views.

HaoLap exploits MapReduce to execute OLAP that leads to the keep query process parallel and efficient In future, we would optimize HaoLap in the following aspect: discuss whether chunks should be compressed, how to compress them, how to perform OLAP on compressed cube, and whether the compression affects the OLAP performance. The other aspect is taking advantage of pre-computation, which is utilized in the work done by Jinguo (You, J.; Xi, J. & Zhang, P., 2008).

E. HaCube

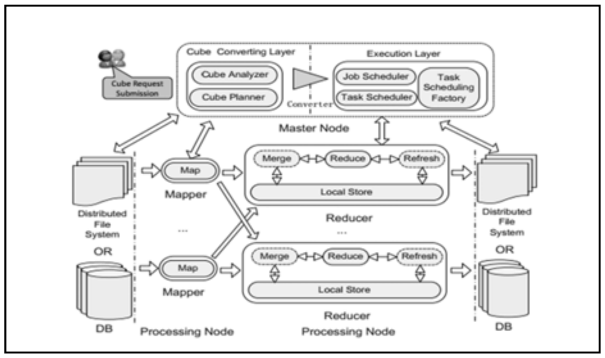

Figure-7 shows an overview of the basic architecture of HaCube. HaCube is an implementations by Hadoop, which is an open source equivalent implementation of MapReduce (Wang, Z.; Chu, Y.; Tan, K.; Agrawal, D.; Abbadi, A, *et al*, 2013). All the nodes in the cluster are divided into two different types of function nodes, including master and processing node. The master node is the controller of the whole system and the processing nodes are used for storage as well as computation.

HaCube inherits some features from MapReduce, such as data read/ process/ write the format of (key, *I*, value, *V*) pairs, sorting all the intermediate data. HaCube enhances with added two optional phases: *Merge* phase and Refresh Phase before and after the Reduce phase (MMRR).

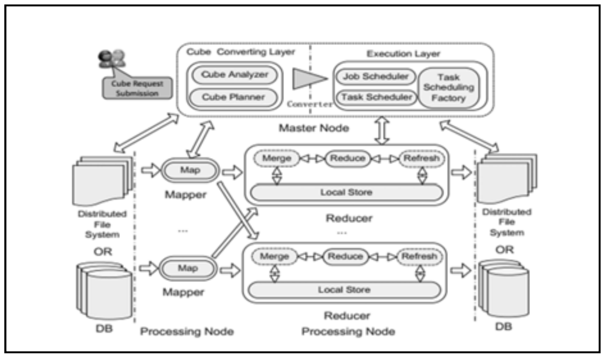

Figure-7. HacCube architecture.

Merge phase has two functionalities. First, to cache the data from the reduce input to the local store.

Second, it is can sort and merge the partitions from mappers with the cached data in the local store. The Refresh phase is developed to perform further computations based on the reduce output. Its functionalities include caching the reduce output data with the cached data in the local store. These two additional phases are intended to fit different application requirements for efficient execution support.

HaCube avoids the aforementioned overheads through storing and reusing the data between different jobs. HaCube, an extension of MapReduce (in term of scalability) and parallel DBMS (in term of efficiency) (Wang, Z.; Chu, Y.; Tan, K.; Agrawal, D.; Abbadi, A, *et al*, 2013).

For the master node consists Cube Converting Layer and Execution Layer. 1. The Cube Converting Layer contains two components: Cube Analyzer, which design to accept user requests and analyze the cube. Next is Cube Planner is developed to convert the cube analysis request into an execution job 2. The Execution Layer function is to manage the execution jobs from the cube converting layer. Three main components that responsibility: job scheduler, task scheduler and task scheduling factory. The job scheduler is same as in Hadoop, which is used to schedule different jobs from different users but we add additional task scheduling factory which is used to record the task scheduling information which can be reused in other jobs. The experimental results show the HaCube performs faster than Hadoop for maintenance. **Error! Reference source not found.** shows the comparison of the exist method.

Table-2. Method comparison.

	MR-Cube	**Tile Cube**	**HaoLap**	**HaCube**
Fault tolerance	Yes	Yes	Yes	Yes
Performance	High	Lower	High	High
Efficient	Yes	No	Yes	Yes
Aggregation	Practical	No	No	Practical
Cost	Low	High	High	Low
Query response	Better	Low	Better	High
Load balancing	Balance	Not Balance	Balance	Good Balance
Cube materializatio n	Efficient	Not Efficient	Efficient	Very Efficient and faster

CONCLUSIONS

This paper has discussed the methods that used in the data cube computation. The data cube is used to represent data along some measure of interest and called a 'cube' because its structure could be 2-dimensional, 3-dimensional, or higher-dimensional and each dimensional will represent the attribute in the database. Based on this introduction, data cube is an easy way to look at the data means allow us to look at complex data in a simple format by using MapReduce. MapReduce is a programming paradigm that allows massive scalability across hundreds or thousands of servers in a Hadoop cluster. The basic term MapReduce actually refers to two separate and distinct tasks that Hadoop perform which are, map and reduce.

HaCube is the latest method that use to Data Cube Analysis because the job scheduler is same as in Hadoop, which is used to schedule different jobs from different users but we add additional task scheduling factory which is used to record the task scheduling information which can be reused in other jobs.

REFERENCES

Anand A. 2008. Scaling Hadoop to 4000 nodes at Yahoo! Retrieved at http://goo.gl/8dRMq.

Dean, J. Ghemawat. 2004. MapReduce: Data processing on large clusters Environment. In: Proceedings of the 6th

Symposium on Operating System Design and Implementation (OSDI'04), pp. 137-150.

DeWitt D. J. and Gray, J. 1992. Parallel Database Systems: The Future of High Performance Database Systems. In Commun ACM, 35(6):85–98, June. http://www.cs.berkeley.edu/~brewer/cs262/5dewittgray92.pdf.

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. 2015. The rise of "big data" on cloud computing: review and open research issues. Information Systems, 47, pp. 98-115.

Hsu I-C. 2013. Multilayer context cloud framework for mobile web 2.0: a proposed infrastructure. International Journal of Communication Systems 2013, 26(5), pp. 610–625.

Jeffrey Dean *et al*, 2004. Mapreduce: Simplified data processing on large clusters. In: Proceedings of the 6th USENIX OSDI, pp. 137–150. https://www.usenix.org/legacy/publications/library/proceedings/osdi04/tech/full_papers/dean/dean_html/.

Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatrao, M., Pellow, F. and Pirahesh H. 199). Data Cube: A Relational Operator Generalizing Group-By, Cross-Tab and Sub-Totals. In: Data Mining and knowledge Discovery1 (1): 29-53(1997). http://arxiv.org/ftp/cs/papers/0701/0701155.pdf.

Li, J., Meng, L., Wang, F. Z., Zhang, W., and Cai, Y. 2014. A Map-Reduce-enabled SOLAP cube for large-scale remotely sensed data aggregation. Computers & Geosciences, 70, pp.110-119.

Li, Z., Yang, C., Jin, B., Yu, M., Liu, K., Sun, M., and Zhan, M. 2015. Enabling Big Geoscience Data Analytics with a Cloud-Based, MapReduce-Enabled and Service-Oriented Workflow Framework. PloS one, 10(3), e0116781. http://ac.els-cdn.com/S0098300414001277/1-s2.0-S0098300414001277-main.pdf.

Nandi, A., Yu, C., Bohannon, P., and Ramakrishnan, R. 2012. Data cube materialization and mining over mapreduce. Knowledge and Data Engineering, IEEE Transactions on, 24(10), pp. 1747-1759. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6104048.

Pramod, P. Prini, K., Aishwarya G. 2015. Map-Reduce for Cube Computation. In IJSRET, ISSN 2278-0882, Vol. 4 Issue 4. http://www.ijsret.org/pdf/121056.pdf Mumick, I., D. Mumick, I. S., Quass, D., and Mumick, B. S. 1997, June. Maintenance of data cubes and summary tables in a warehouse. In ACM Sigmod Record , 26(2), pp. 100-111). ACM. http://dl.acm.org/citation.cfm?id=253277.

Song, J., Guo, C., Wang, Z., Zhang, Y., Yu, G., and Pierson, J. M. 2015. HaoLap: a Hadoop based OLAP system for big data. Journal of Systems and Software, 102, pp. 167-181.

Wang, Z., Chu, Y., Tan, K. L., Agrawal, D., Abbadi, A. E., and Xu, X. 2013. Scalable data cube analysis over big data. ArXiv preprint arXiv:1311.5663. [cs.DB].

You, J., Xi, J., and Zhang, P. 2008. A parallel algorithm for closed cube computation. In: Seventh IEEE/ACIS International Conference on Computer and Information Science. ICIS 08, pp. 95-99.