Concept Paper on MapReduce In Comparison With DBMS

Shrija Chavan

School of Applied Technology

Department of Information Technology and Management.

Illinois Institute of Technology

Table of Contents

Abstract

This article is an attempt to provide an overview on the concept of MapReduce in comparison with Relation Database explained by authors David J. DeWitt, Michael Stonebraker and Mark C. Chu-Carrol. To start with the brief understanding of what MapReduce is, followed by the discussion of the drawbacks of MapReduce are over DBMS are explained by David J. DeWitt and Michael Stonebraker. But, with the support of existing literature review and his personal experience on MapReduce, author Mark C. Chu-Carrol represents his ideas about the MapReduce. Finally, in the conclusion the author's viewpoint on MapReduce is presented with examples.

Keywords: MR- MapReduce, DBMS-Database Management System.

Introduction

The word “MapReduce” is now extensively used in the todays IT world and having said that, it also happens to be the one of the major concepts in BIG DATA analysis. There have been many articles written on MapReduce but, in this concept paper we are mainly going to compare the ideas and experiences of authors David J. DeWitt, Michael Stonebraker and Mark C. Chu-Carrol who have completely different standpoint about MapReduce. According to David J. DeWitt and Michael Stonebraker, MapReduce is a giant step backward and they explain their viewpoint based on their research and experiences. However, author Mark C. Chu-Carrol totally disagree with the statement and explains that Databases are like hammer and MapReduce is like screwdriver by giving his personal experiences to support his thoughts. This paper will discuss the MapReduce concept from the viewpoint of both the authors and make a conclusion based on today’s analysis of MapReduce.

MapReduce A Major Step Backward

In January 2008 author's David J. DeWitt and Michael Stonebraker being both educators and researchers made this statement that "MapReduce is a major step backwards". This was the time when cloud computing was just introduced and many people were not aware of this term MapReduce. The authors by their experience and knowledge said that MapReduce may be a good idea for writing certain types of general-purpose computations, but to the database it is step backwards. They supported their point of view by explaining the below points:

1.**A giant step backward in the programming paradigm for large-scale data intensive applications**: Having said the above statement, the authors explained that MR do not have any of the following techniques which the database has learned from the last 40 years.

Schemas: DBMS follow the structure of schema by defining the datatypes and the fields which are inserted into the database which makes sure that no junk data is going into the database. However, the MR do not follow any of these rules and takes in all kind of data which can corrupt the entire dataset.

Separating the schema from the application: In DBMS, the application is separated from the scheme(database) for the reusability of the code or to do any modifications to the application if the changes are needed. MR do not have such concept, the application and the schema are same in MR where changing the code becomes a tedious job. The developer must go through the entire application to find out the source code which is not a diligent work and is time consuming.

Using high level programming language: The authors explained that a high-level programming language should be followed by all the application developers since it's easy to write, understand and learn as compared to the low-level programming language used MR.

2.A sub-optimal implementation, in that it uses brute force instead of indexing:

According to the authors, DBMS uses Indexes to search the data faster and to apply filters to the dataset to the get the results quicker. Also, DBMS has a query optimizer to decide whether to use index or to use brute force sequential search. However, MR lacks the indexing technique and only make use of brute force sequential search, this being an disadvantage of MR. The authors claim that the technique which MR uses i.e. the grid computing was already explored by DBMS in late 90's and was not a very new idea. Besides that, DBMS has high-performance, commercial, grid-oriented SQL engines (including schemas and indexing) for the past several years. MR fails to have these advantages.

They also doubt on the MR's ability to access data faster than SQL because they claim that producing 100s of map and reduce instance will fail when two map nodes try to read the same input file simultaneously, will induce large numbers of disk seeks and slows down the effective disk transfer rate.

3.**MapReduce is not novel at all:**

The authors say that MR is not the first one to introduce new paradigm for processing large data sets. The strategies utilized by MapReduce are over 20 years of age. Partitioning a substantial informational index into littler segments was initially proposed in "Utilization of Hash to Data Base Machine and Its Architecture" as the reason for another kind of join calculation.

Teradata has been offering DBMS using these methods for over 20 years; precisely the strategies that the MapReduce claim cases to have invented.

4.**Missing most of the features that are routinely included in current DBMS**. The authors gave a list of techniques which the MR lacks as compared to DBMS:

Bulk Loader: Transforms the data into desired format and transfers into database

Indexing: Used to find the results quickly, avoids going through the entire dataset.

Updates: Alterations in the database.

Transactional: to support parallel update and recovery from failures during update

Integrity Constraints: To keep the unwanted data away

Referential Integrity: To keep the unwanted data away

View: Schema can be changed by just making alterations to the application and not database.

In outline, MapReduce gives just small percentage of advantages as compared DBMSs.

5.**Incompatible with all the tools DBMS users have come to depend on:**

MR is said to be incompatible with the DBMS tools because the following tools available with DBMS and not in MR.

1.Report Writers

2.Business Intelligence tools

3.Data mining tools

4.Replication tools

5.Database design tools

In summary, the authors try to convince that DBMS has more advantages over the modern MR. They affirm not to neglect the lessons of over 40 years of database innovation as compared to the MR which has no history or innovation compared to DBMS.

Databases are hammers and MapReduce is a screwdriver

In contrast of what was said by David J. DeWitt and Michael Stonebraker, Author Mark C. Chu-Carroll on January 22nd, 2008 replied by completely disagreeing with all the statements made by David J. DeWitt and Michael Stonebraker about MR.

Being the employee of Google, he explains that his ideas are completely personal and has no relations to being an employee of Google. He also cleared that he has never used MR at his work. He explained that the work done by him during his PHD was very much like MR and hence he supports MR.

According to him, "MapReduce is a library that lets you adopt a stylized way of programming that's easy to split among a bunch of machines." He says that the basic idea of MapReduce is to split the task to different machines to get quicker results. The map splits the job into several pieces and send it to different machines to run at the same time, then reducer then collects all the results from the subparts and combines them together to present as a single answer. He also explains that MR can be best parallel programming language ever as its very easy to write.

Further, he explains why he feels that DBMS is nothing more than a most beautiful, wonderful, perfect hammer in the whole world, he explains that people who believe in RDBMS think that everything other than RDMS is like a nail in from of the hammer(DBMS). He claims that DBMS people have started believing in it so much that they don't accept anything which is better than DBMS.

Moreover, he also said that DBMS is one of the best tools he has ever used and can do brilliant things and make amazing software. But, it cannot be the replacement for everything. He

says that RDBMS cannot handle non-tabular data, they are known to do poor job on the data structures and MR is not a replacement DBMS.

Being said that above things about databases, he completely slashes all the 5 points made by David J. DeWitt and Michael Stonebraker, and explains his point of view towards MR.

1.A giant step backward in the programming paradigm for large-scale data intensive applications: Author Mark C. Chu-Carroll claims the above sentence to be nothing kore than rubbish and says that MR is not a replacement for relational database.

He says that when you have a structured data which can be run on a single machine and can give quicker results, one should use databases and not MR. But, when you have large unstructured data that takes a lot of time to run on the single machine, then you should use MR. He also says that MR is not a relational application like databases which works on rows and columns of the tables in the database. MR takes in the input as the large dataset which are just rows of data in a file and start working on it. He clearly explains the difference between MR and DBMS and shows that they both are not comparable as both are used for two different things.

2.MR is a sub-optimal implementation, in that it uses brute force instead of indexing:

He explains the above point and says that indexing is only useful when your data is tabular. But, when you are working on large datasets which are not fundamentally relational and needs computations, then indexing is not going to help. The basic idea of MR is to write a program which can run on a large dataset in a reasonable amount of time, indexes are not useful if your task needs a lot of computation on a huge dataset.

3.MR is not Novel: The authors say that MR never claimed to be Novel and is not supposed to be novel. It was inspired by the functional programming language using map and reduce primitives.

It was created to form a data parallel computing which can be scalable implementation of a well understood parallel programming.

4.Missing most of the features that are routinely included in current DBMS:
Author again say the same thing which he said in the first point that, MR is not a database application and they should stop comparing both together. He tells that when you have relational database problem, use relational database, and when you have an issue of massive computation tasks, and wants the job to be run on different computers or machines then use MR. Databases are not designed to do massive computations, they don't work on them at all.

5.Incompatible with all the DBMS tools: Author Chu again says DB developers do not accept anything other than relation database tools. No database tool can work in place of a MR program. Instead a good database tool can turn out to be a very poor MR tool. MR is not meant to replace database tools. They both are for a completely different use and should not be compared.

Discussion

After carefully reading and examining the above two articles by David J. DeWitt, Michael Stonebraker & Mark C. Chu-Carroll, I conclude where I believe in all the view- points presented by Mark C. Chu-Carroll. By reading these two article, it is very clear that DBMS and MR are two different things and be compared on a same platform.

"MapReduce is a good fit for problems that need to analyze the whole dataset in a batch fashion, particularly for ad hoc analysis. An RDBMS is good for point queries or updates, where the dataset has been indexed to deliver low-latency retrieval and update times of a relatively small amount of data. MapReduce suits applications where the data is written once and read many times, whereas a relational database is good for datasets that are continually updated" – (Reference Chapter 01 OReilly.Hadoop.The.Definitive.Guide.4th.Edition.2015).

The above statement explains many points stated by author Mark C. Chu-Carroll in his

discussion. The below table explains in brief the major difference between DBMS and MR, and conclude how both have their own advantages and disadvantages:

	RDBMS	**MapReduce**
Data size	Gigabytes	Petabytes
Access	Interactive and Batch	Batch
Updates	Read and write many times	Write once, read many times
Transactions	ACID	None
Structure	Schema-on-write	Schema-on-read
Integrity	High	Low
Scaling	Nonlinear	Linear

Fig 1: Comparison between MapReduce and RDBMS

By looking at the above fig1, we can clearly say that MapReduce is fundamentally a batch processing system and cannot be used when we need quicker results, it's not interactive as relational databases.MR should be used only when you have no hurry of the results because it runs on a very huge dataset compared to DBMS. They fall under the distributed computing and large scale processing data structure. But as the Hadoop terminology is getting older, many of the short coming of Hadoop are taken care. Now, Hadoop has introduced many more applications like MR such as YARN and HIVE, IMPALA which try to recover some of disadvantages of MR such as Interactive SQL, iterative processes, stream processing, search etc.

Moreover, Hadoop now provides an API to MR that allows you to write your map and reduce functions in languages other than Java. So, you can use any language that can read standard input and write to standard output to write your MapReduce program, this makes MR to stand one step ahead of Relational database which can only make use of SQL to query the data.

Conclusion

Based on the authors experience and research in RDBMS and MapReduce, I conclude that relational database and MapReduce are two different concepts and are not interchangeable. They both can be used interchangeably by looking at the nature of data such as the size of the dataset, time required to process the data, structured or unstructured data and so on. There are situations where relational databases can be more efficient than MR and vice versa. Both can be great tools for computation of a task, but the approach is different. However, the growing demand of managing big data make the MR a better platform than relational databases. It is also overcoming some of its short coming compared to SQL to query the data in a short period of time. So, if given a chance to pick one of them, I would rather pick MapReduce over relational databases as it can fulfill the demands of MR as well as relational databases.

References

1. http://scienceblogs.com/goodmath/2008/01/22/databases-are-hammers-mapreduc/
2. http://homes.cs.washington.edu/~billhowe/mapreduce_a_major_step_backwards.html
3. https://static.googleusercontent.com/media/research.google.com/en//archive/gfs-sosp2003.pdf
4. https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf
5. OReilly.Hadoop.The.Definitive.Guide.4th.Edition.2015.
6. Fig 1:(Reference- chapter 01-OReilly.Hadoop.The.Definitive.Guide.4th.Edition.2015).