MapReduce – How it Evolved

Aniruddha Erande

Illinois Institute of Technology, Chicago

Abstract

DBMS systems have been in the market for a very long time, but with the introduction of new framework ‘MapReduce’, there has been comparison between the two techniques, and criticism on both. With the introduction of MapReduce, it lacks many features that DBMS systems already provide. But MapReduce is totally different from DBMS as MapReduce does not deal with relational data.

Keywords: DBMS, RD, relational, MapReduce, brute force, indexing.

MapReduce – How it Evolved

The following paper will discuss about how MapReduce became so popular in comparison to Relational Database and its ups and downs, taking reference from two papers – one paper from Washington university[1], and the other from a Science Blog[2].

Summary

There was a time when Relational Database was the king of databases, useful in handling big data, although not in a very effective way, but after the introduction of MapReduce framework, there has been a fierce competition between RDB and MapReduce. There are some common points that both the papers discuss, and they will be elaborated and compared below:

1. A giant step backward in handling big data applications

The key point states that the MapReduce framework represents the time in 1960s before DBMS were not in existence. The Washington paper suggests that DBMS follows the same schema overall and it is obeyed through the whole process. This leads to the application not barging "garbage" into the data set. By this way, the storage is not filled with all unwanted data. On the other hand, MapReduce does not have any control on garbage disposal. Even a corrupted MapReduce dataset will break down the whole system.

The Science blog paper totally opposes the point. The blog states that if the data is labelled as relational, and which is feasible to run on a single machine, then RDB is the key. But if there is a large-scale data, which takes some significant amount of time to solve on a different machine, then MapReduce should be used, which distributes the computation along different systems and computes parallelly.

2. MapReduce works with brute force technique instead of indexing

In the Washington paper, MapReduce is compared with ongoing DBMS systems where DBMS uses indexing to parse data for search whereas MapReduce performs brute force technique. According to them, indexing is far way better than brute force option. Also, a point to note is MapReduce works parallelly among a grid of computers. This parallel execution was discovered in the DBMS in 1980s, so this is not something new that MapReduce has implemented.

The Science blog defends MapReduce by stating that if there is a tabular data, and if there is a central index, then indexing will work pretty good for RDB. But when there is really complex data with floating point multiplication, then indexing won't work. Here, MapReduce will help to work swiftly.

3. The technique used takes us back 25 years

According to the Washington paper, MapReduce is not a novel framework, and practices the technique of partitioning a large data into small modules and executing it. This idea was first implemented in "Application of Hash to Data Base Machine and Its Architecture"[3] as the basis of generation of new algorithm.

According to the Science blog, again it opposes the point saying that MapReduce never claimed to be a novel technique; and it is *not* novel. MapReduce works in two ways – Map & Reduce. Map is used to divide the data, execute it, and Reduce gathers all the data into one output. MapReduce is basically about parallel computing.

4. Missing some techniques that are imposed in DBMS

The Washington paper states that MapReduce is clearly missing some features like:

- Bulk loader – loading data into the database as the whole,
- There is no indexing,
- There is nothing to update the data in the database. In MapReduce, first the data have to deleted and then add the updated data,
- No Transaction failure recovery with parallel computing,
- No mechanism to collect and evacuate garbage data,

According to the Science blog, these all points are rubbish. MapReduce is just another version of relational database, and it is good. MapReduce isn't a database application. DBMS was not build to handle big data and parallel computing, MapReduce is!

5. Incompatible with DBMS

According to the Washington paper, MapReduce will be difficult to use with end-to-end task, and does not possess the following class of tools:

- No report writing,
- MapReduce does not contain Business Intelligence tools,
- No Data Mining tools,
- No Replication tools,
- No assistance in building a database,

The Science blog states that MapReduce is not a relational database. So, it will not possess any of the qualities pointed above. Database-oriented tools will not work for MapReduce.

Conclusion

So finally, to infer, MapReduce is different from all DB systems and it is what it is. It should not be criticized, because it is great in handling a large amount of data, executing it parallelly on different computers, despite some features not user friendly. So, I support whatever the Science blog suggests, and accept the fact that MapReduce will inherit whatever is missing in the future.

References

[1] Chu-Carroll, Mark C. (January 22, 2008). Good Math, Bad Math. *Databases are hammers; MapReduce is a screwdriver*.

[2] DeWitt, David (January 17, 2008). The Database Column. *MapReduce: A major step backwards.*

[3] Kitsuregawa, Masaru, Tanaka, Hidehiko, Moto-Oka, Tohru (1983). Application of Hash to Data Base Machine and Its Architecture. 63-74.