MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

A Comparison of Approaches to Large-Scale Data Analysis

Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, Michael Stonebraker

●●●

Dayna Eidle
Big Data Paper Summary
Database Management
Due: October 30, 2017

MapReduce - Main Idea

- Responds to the problem of not being able to “parallelize the computation, distribute the data, and handle [system] failures”
- Generates and processes large data sets across multiple machines
- MapReduce “hides the details of parallelization, fault-tolerance, locality optimization, and load balancing” making it easy for anyone to work with.
- A large number of problems and computations can be solved using MapReduce.
- This system actually reduces the amount of data sent across a network, therefore saving network bandwidth.

MapReduce - Implementation

- MapReduce is split into two user-designed functions — *map* and *reduce*
    - The *map* function takes a set of data and splits it into key-value pairs. These key-value pairs can be anything the user needs for their specific system (i.e. word in a document, and iteration number of that word in the document).
    - The *reduce* function essentially summarizes the map data. Using the example from above, it might output the total number occurrences of each word in the document.
- It makes copies of the user program, designating one as the master and the rest as workers
    - The master assigns work to the workers and keeps track of and oversees the work they are doing
    - Workers are assigned a map task and then must read the contents of the input split, parse the input data into key-value pairs, and then pass those pairs into the *map* function.
    - Reduce workers sort through these key-value pairs and then pass them along to the *reduce* function.
- Final outputs from *MapReduce* are then put into an output file.

MapReduce - Analysis

- MapReduce provides and easy-to-use program for dealing with big data. The fact that Google was able to simplify the process of sorting and storing big data is interesting in itself. However, I also found it interesting that this method can be used to solve a number of various computational problems. It seems like it would be extremely useful and efficient because it reduces the amount of data being sent across the network. Although it seems very versatile, it restricts the programming model, as mentioned in the paper. This means that even though it can be used in different forms, there is very little that can be done with the data outside of the MapReduce function.

Comparison of Approaches - Main Idea

- A comparison and evaluation are made between MapReduce and parallel SQL database management systems.
- Any parallel processing task can be dealt with using database queries or MapReduce tasks.
- Overview of the results:
    - SQL database management systems are faster and require less code
    - MapReduce loaded the data faster
    - It depends on the circumstances for which one is “better” to use

Comparison of Approaches - Implementation

- They used Hadoop (version 0.19.0), DBMS-X, and Vertica to test the differences in performance level between MapReduce and parallel SQL database management systems.
- Tasks used for testing:
  - Grep task - original MapReduce task
    - This tested for data loading efficiency and overall task execution
  - Analytical tasks (related to HTML document processing)
    - This was broken down into five parts: data loading, selection task, aggregation task, join task, UDF aggregation task

Comparison of Approaches - Analysis

- In general, the authors came to the conclusion that both parallel database systems outperformed Hadoop in the given tasks.
    - Although tested on 100 nodes, they feel that they even with 1,000 nodes the results would have been the same.
- Hadoop proved easier to set up than the database systems and was also less expensive
- Both methods had pros and cons to them, and even though the parallel database systems performed the best, it can depend on what each user feels is necessary for for performance for a database. The authors believe that there is still a lot to learn from both systems.

Comparison of the Two Papers

- Having the first paper I read be specifically about MapReduce made reading the second paper more interesting. In the MapReduce paper, the authors talked about all of the positive points about the MapReduce system. At the time, Google was using that system to organize their database. The authors continued to mention how it could be used for different computations and how the simplicity of it made it something worth having (with the tone that their implementation was the best).
- Although the comparison paper did highlight the simplicity and some of the other practical features of MapReduce, it talked more about how parallel SQL database management systems outperform MapReduce. This paper brought out more of the cons of using MapReduce. Interestingly enough, the comparison paper also mentioned how Google is no longer using MapReduce which is somewhat comical considering how they discussed it in the MapReduce paper.

Michael Stonebraker Talk - Main Idea

In his talk, Michael Stonebraker emphasis the fact that one size does not fit all. In fact, one size fits none. He focuses in on how row-based relational database systems have already begun to be phased out and will soon be out of the picture completely. He goes through different markets and talks about the database structure of each one. He continues by saying that traditional relational database management systems don't have a place or are the best choice to use in any of those markets anymore. Stonebraker believes that with the amount of processor diversity coming out that we should be seeing a good amount of database management system implementations in the near future. In his closing, he says that he feels like this new era and new ideas is really a good thing because it forces us to move away from the "one size fits all" idea.

MapReduce vs. Comparison of Approaches and Stonebraker

- Advantages
    - As discussed in the comparison paper, MapReduce is simpler than most systems and is also less expensive. Also, with MapReduce not being row-based, I would like to think that Michael Stonebraker would be relatively fond of it. Based on his talk, it would seem that there might be *some* place for it in the world of database management systems.
- Disadvantages
    - While the authors of the MapReduce paper sung the praises of MapReduce in the paper, the comparison paper quickly showed that there are better management systems out there. Parallel SQL database management systems outperformed MapReduce in a series of tests, showing the weaknesses of MapReduce and the strengths of other options. Going along with the whole theme of Stonebraker's talk, although MapReduce is not row-based, we still can't be quick to assume that it is the best thing out there (comparison paper showed that it's not) just because it is a new and simple system.