Julian Singkham
CS 2300/11
Dr. Berisha

Paper Review 2: Map Reduce

The purpose of this paper is to explain how Google's implementation of the MapReduce algorithm works on large clusters of data, the experience gained from using MapReduce, and how it compares to other models. Sections 2-4 explains how programming with MapReduce works with examples, how parallelism is achieved, and how it handles faults. Section 5 covers the performance of MapReduce and illustrates execution times with different parameters such as the use of backups and machine failures. Section 6 discusses the experience gained from using MapReduce such as explain how MapReduce surpasses its predecessor. Section 7 goes over alternatives to MapReduce and details the benefits and differences between them.

1. Describe an example real-world computational operation that can use MapReduce.

I believe advertisements are a great application for MapReduce. The human psyche is highly complicated, and each person's likes and dislikes are difficult to quantify and gauge. By analyzing what a user views, the purpose behind their action, and how long a user views something can be used to create targeted adds that are relevant to the user. With the use of tagging and relationship between data, we can create a system that can pinpoint a person's interest and what they're likely to buy. For example, lets say Bob very much likes sports, football in particular and spends a lot of his time watching football games and viewing football related merchandise. Based on data gathered from Bob, products with the football tag should be advertised to Bob. However, for football people tend to be polarized to one team and greatly dislikes the other teams. This is where relationship between datapoints come into play by displaying products related to Bob's favorite football team. As social creatures we consume a very large amount of media, and to keep track of what a user likes and dislikes requires a large amount of data, particularly a very large amount of data when viewing the entirety of mankind. MapReduce would be able to do this with highly efficient performance and data usage.

2. Why are ordering guarantees necessary?

Ordering guarantees are necessary as they allow sequential (queue) processing of data, meaning that older intermediate key/value pairs are processed before newer intermediate key/value pairs. This also aides in calculating how long it takes a job to complete since the system is run like a queue. Lastly for debugging purposes, by having a sorted history of job completions, we can pinpoint where the problem occurred and what processes were affected.

3. Why has the MapReduce algorithm/implementation presented been successful?

MapReduce implementation has been successful due to its high level of performance, reliability, and simplicity. Performance is achieved by its very high level of parallelism where jobs are split between clusters of potentially thousands of map workers which get directly sent to reducers. The high level of parallelism also allows for high levels of reliability as the same job can be calculated on multiple machines. Additionally failures are easily trackable and all affected systems can have their individual jobs restarted. Finally, since parallelism is under the hood so to speak, the user can easily use the MapReduce algorithm without needing to know how the system works. By removing a potential barrier to entry, many data analysts, from amateur to experts, can use this system; thus aiding the community as a whole.

4. Describe at least one potential challenge/drawback of this approach.

One major drawback to this approach is the fact that MapReduce is a batched process, meaning the data must be owned at the time of computing. This means that streaming data is unable to utilize MapReduce as the algorithm needs to wait for the stream to end before acting on it.

5. What is the hardest to understand part of this paper?

Section 3, implementation, was the hardest part of the paper to understand as it was very word heavy. As a visual learner, I found that figure 1 explained more of the MapReduce operation than what was written. To me the process of error correction was difficult to understand at first as I had to draw out the steps MapReduce uses to analyze which tasks are affected by the error and how it seamlessly restarts the applicable jobs.