Pregel: A System for Large-Scale Graph Processing and A Comparison of Approaches to Large-Scale Data Analysis

By: Jake Litts

Main Idea of the Pregel Paper

- The scale of large graphs poses challenges to their effective processing.
- In order to combat these challenges, the system Pregel was created.
- This resulting API provides a framework for processing these large graphs, while also being expressive and easy to program.

How This Idea is Implemented

- Designed for the Google cluster architecture, where each cluster consists of thousands of commodity PCs organized into racks with high intra-rack bandwidth.
- Fault tolerance is achieved in Pregel through checkpointing; worker failures are detected through regular “ping’ messages that the master issues to the workers.
- Confined recovery is an idea under development that will improve the cost and latency of recovery.
- Worker machines maintain the state of their own portion of the graph.
- The master is primarily responsible for coordinating the activities of the workers.
  - Knows all alive workers, has operations terminated a barriers, and maintains statistics about the computation and state of the graph.

My Analysis

- What intrigues me the most about Pregel is how it is able to take large graphs of data and break it down into manageable pieces for each individual worker computer to handle. Meanwhile, the Master computer keeps track of which worker computer are working, how to disperse the work, and how to clean up any messes the Workers might run into.
- The implementation of Pregel also allows four real-world problems to be solved with simplified versions of existing algorithms.
  - These problems are: Page Rank, Shortest Paths, Bipartite Matching, and a Semi-Clustering algorithm.
- This is important because the simpler the algorithm is, the quicker it will run.

Main Ideas of the Comparison Paper

- MapReduce, called a dramatically new computing model, works much faster than the previous system used.
- The authors of this paper argue that the speed of a given process should not be the defining factor to choose MR (MapReduce) over parallel SQL DBMS (database management systems).
- The authors prove that even though parallel SQL DBMS is not as fast, the observed performance was strikingly better than that of MR.
- In their experiment, the authors of the paper concluded that the SQL DBMS were significantly faster and required less code to implement each task, it took longer too tune and load the data.

How These Ideas are Implemented

- The authors first explain the advantages of each large scale analysis approaches: MapReduce and Parallel DBMSs.
  - MapReduce is attractive due to be simplistic, as it only consists of two functions.
    - The Map function reads a set of "records" and splits them into disjoint buckets. Each Map instance is designed a different distinct portion of the input file by the MR scheduler to process.
    - The Reduce function processes or combines the records assigned to it in some way, and then writes records to an output file (in the distributed file system), which forms part of the computation's final output.
    - The MR scheduler only needs to decide how many Map instances to run and how to allocate them to available nodes, as well as decide on the number and location of nodes running Reduce instances. The MR central controller has to coordinate the system activities on each node.
  - Parallel DBMS have two key aspects that enable parallel execution: most (or even all) tables are partitioned over the nodes in a cluster and that the system uses an optimizer that translates SQL commands into a query plan whose execution is divided amongst multiple nodes.
- The authors then pit the performances of the MR model against the parallel DBMSs.
  - This is done through 5 tasks: The Original MR task ("Grep task"), the Data Loading task, the Selection task, the Aggregation task, and the Join task.
- After conducting their tests, they discussed possible reason for their results.

My Analysis

- To me, I see clear advantages to each analysis process.
- On one hand, if you need to process larger data, MR seems like the better option.
  - It seems to me that though it takes longer to set up processes in MR, the speed it takes to complete processes vastly outweighs that of SQL DBMS.
  - The trade off is that the it is not as accurate as SQL DMBS. However, you are able to complete a larger number of processes than the later.
- On the other hand, SQL seems to be the better option when working with smaller data. Though slower, it is able to be more accurate.
- It makes sense to me that we are using SQL DBMS for this class rather than MR due to the fact that we are looking for accuracy in smaller queries, rather than working with larger data.

Comparison of the Two Papers

- In the Pregel paper, the system of Pregel was being praised as being the faster alternative to processing large graph data.
  - After reading this paper alone, it seemed that the only way one would want to tackle the problem of dealing with larger data would be to use a system such as Pregel.
- However, the second paper states that if you want larger data to be processed more accurately, parallel SQL DBMS are a better option.
  - The problem with parallel SQL DBMS is that though the data will be processed more accurately, it will also take much longer to process.

Main Ideas of the Stonebraker Talk

- For the span of years between 1970 and 2000, Relational Database Management systems were the answer to any question possible.
  - Later realized in 2005 that this would never practically work.
- In 2015 they found out that all of the previous RDBMS like Oracle, DB2, etc. are obsolete in every major market.
- In each major market, they are oriented toward a specific application that is different from each other.
  - The traditional RDBMS are not good at any of these markets.
- Now there is greater room for new opportunities of ideas.

Advantages and Disadvantages of Pregel

- After watching the Stonebraker video, it seems to be that Pregel has a step up on using a SQL DBMS due to the fact that it is specialized to the task of processing large graphs.
  - In this instance, I assume that there is no SQL DBMS that can do this job better.
- Another advantage to Pregel is that it can process this large graphical data quickly; this is especially the case when comparing it to SQL DBMS.
- One disadvantage that I assume to be the case after reading the second paper is that it might not be as accurate than if you used a SQL DBMS.
- Another disadvantage of Pregel is that it is becoming a piece of production infrastructure for its user base. This means that the creators are no longer at liberty to change the API without considering compatibility.
  - However, they believe that the existing programming interface is sufficiently abstract and flexible to be resilient to the further evolution of the underlying system.