Pregel: A System for Large-Scale Graph Processing

Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski

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

Pregel: Main Idea

- Processing large data models in fractional time with the use of a graph algorithms.
- Processing problems such as Web 2.0, transportation routes, disease outbreaks, etc.
- Inspired by Valiant's Bulk Synchronous Parallel Model.
- Distributed as a directed graph, and computed by supersteps.
- Pregel programs are free of deadlocks, and do not face data races.
- Minimize latency by delivering messages in asynchronous batches.
- All computations are performed locally, only messages are sent via the network.

Pregel: Implementing

- Built for optimization on the Google cluster architecture.
- Graphs are broken up and partitioned into groups of outgoing edges. Partitions are later assigned to worker machines, allowing one machine to process many partitions in parallel.
- Asynchronous messaging allows for messages to arrive before the superstep is finished, overlapping, communication, and batch processing.
- Checkpoints allow for fault tolerance. The amount of checkpoints is determined by checkpoint cost versus recovery cost.
- High bandwidth allows for large graphs to be processed by a geographically distributed personal computers in parallel.

Pregel: Analysis

- The idea of checkpoints between supersteps allows for supersteps to be reloaded and reprocessed in the event a superstep is missed.
- Shortest path is a problem faced in many graphs. Pregel's ability to allow for each vertex to communicate the minimum distance from itself to the source vertex, then compare these distances and pass the value onto the next vertex allows for the algorithm to choose the shortest path more frequently.
- Semi-clustering is a technique used to process social graph data which is becoming exceedingly more frequent. This is useful in determining the social interactions between one person and multiple persons. This can be found extremely useful in obvious social networks on Web 2.0.

Comparison

- As far as a comparison goes it is important to note that MapReduce forces its entire map data to be pushed between steps, this can be stressful to the system, and cause a significant amount of communication and overhead. Pregel does not, Pregel only utilizes network transfers for messaging. Pregel's supersteps avoid the programming complexity caused by MapReduce. Pregel seems to be more useful on a large number of clustered computers and more advanced in it's ability to pass values throughout vertices and edges.
- Both DBMSs and Pregel utilize parallelism by partitioning data across a number of independent computers.
- Pregel is significantly more fault tolerant than DBMSs, due to it's check pointing abilities. DBMSs will not checkpoint during queries, if a query fails it must be restarted.

Advantages/Disadvantages

- Pregel has great advantage to DBMSs in fault tolerance due to its ability to checkpoint between supersteps.
- The first paper (Pregel) tests on 300 multicore PCs, whereas the second paper runs test cases on 100 multicore PCs. In the second paper MapReduce which is like Pregel shows significant performance over the DBMS when processing datasets, however is crushed in performance in many other tests.
- DBMSs are able to process queries faster, and more accurately.
- Hadoop (MR like) systems are easy to implement but maintenance may lead the user to prefer the apparent DBMS learning curve.
- It seems to be that DMBSs are much more accurate, and in modern day the solution of choice, however Pregel (I assume) will preform better in Web 2.0 scenarios such as social networking maps.