Big Data Paper Summary

Kevin Callahan

May 9th 2014

A Comparison of Approaches to Large-Scale Data Analysis

Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, and Michael Stonebraker. 2009. A comparison of approaches to large-scale data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data* (SIGMOD '09), Carsten Binnig and Benoit Dageville (Eds.). ACM, New York, NY, USA, 165-178. DOI=10.1145/1559845.1559865 http://doi.acm.org/10.1145/1559845.1559865

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. 2010. Pregel: a system for large-scale graph processing. In*Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*(SIGMOD '10). ACM, New York, NY, USA, 135-146. DOI=10.1145/1807167.1807184 http://doi.acm.org/10.1145/1807167.1807184

Main Concepts of Pregel

- Created a computational model for Large-Scale Graph Processing
- Important need for efficient processing of the large graph
- Designed to fill a need for a platform for implementing an algorithm to process such a graph
- Required to be:
  - Scalable
  - Fault-tolerant
  - Flexible API
- Executed through message passing

How Pregel is Implemented

- The graph is initialized from input data, sourced from multiple possible file formats
  - Input type separate from the graph computation
- Computations expressed as a sequence of iterations, or supersteps, separated by global synchronization points
  - Within a superstep the vertices compute in parallel, each executing the same function
- Vertices can receive previously sent messages, send messages, modify its own state and that of its outgoing edges, or mutate graph topology
  - This allows for a wide range of algorithms to be expressed
- Computations continue until a vertex votes to halt, at which point it does not execute unless reactivated externally
- The program or algorithm as a whole terminates when all vertices are simultaneously inactive with no messages in transit

Analysis of Pregel Implementation

- Easy to code for
  - “Think like a vertex”
  - Flexible API
  - Developed with user input
- Good Fault Tolerance
  - Checkpointing at each superstep
  - Worker failures moderated by a master
  - Basic Recovery - current workers reload to the most recent available checkpoint
  - Confined recovery - uses outgoing message logs to allow recovery to only lost partitions, using log messages from healthy partitions and recalculating, the system can recompute to current superstep for just lost partitions

Comparative approaches

- Fault tolerance of Pregel can be more effective than traditional relational systems, as an interrupted transaction must be completely reversed and then reattempted, while Pregel can recover and continue
- Schema not as strongly implemented as the traditional relational model, any schema implemented must be user defined
- Effective for use with a small programming team, this is supported by tools such as single machine mode, which aids in rapid prototyping
- Requires much more coding than the traditional relational model

Advantages and Disadvantages of Pregel

Pros

- Fault tolerance excellent, especially ability for confined recovery
- Very flexible, allows for large expression
- Familiar for users experienced in procedural languages

Cons

- Requires a large amount of user definition and user written code
    - There are defaults however large amount of tuning needed, based on the algorithm being expressed
- Any indexing needs to be implemented by the user