MapReduce

Anders Lykkehoy

October 25, 2017

Bibliography

- Dean, Jeffery, and Sanjay Ghemawat. “MapReduce: Simplified Data Processing on Large Cluster.” OSDI'04 Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation, vol. 6, 8 Dec. 2004.
- Pavlo, Andew, et al. “A Comparison of Approaches to Large-Scale Data Analysis.” SIGMOD '09 Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, 29 June 2009, pp. 165–178.
- “Michael Stonebraker on his 10-Year Most Influential Paper Award.” Brown Database Group, 2015.

MapReduce

- Abstract way to parallelize code for large data sets
- Able to easily distribute workload across hundreds to thousands of machines
- Hides messy details of parallelization through a simple interface
    - Distribution of data
    - Fault-tolerance
    - Load balancing

MapReduce Implementation

- Takes set of input key / value pairs and produces a set of output key / value pairs
- First user defined function (Map):
    - Input key / value produces an intermediate key / value
- Second user defined function (Reduce):
    - Takes the intermediate key / value, merges the values to form a smaller set of values
    - Typically down to a single output value per reduce

MapReduce Analysis

- Because it hides the implementation it is easy to use
- Scales very well with more machines
  - Given ~1Tb of data, 1700 machines completed in under 600 seconds
- Practical:
  - Already in use for Google’s web searching
  - Used in machine learning problems

Comparison Paper

- Parallel Databases vs. MapReduce
- Parallel Databases:
  - Database systems running on nodes in clusters
  - Translates SQL into query plans
  - Those plans are divided into the nodes
  - Transparent to the user

Implementation

- Tested Systems:
  - Hadoop: MapReduce
  - DBMS-X: Parallel Database
  - Vertica: Parallel Database where data is stored as columns
- In almost every test, the Parallel Databases out performed MapReduce
  - Grep, Aggregation, Join, Selection

Analysis

- The paper only tests on up to 100 nodes
    - In the original MapReduce paper tests were conducted using over 1000 nodes
    - Could be the cause of MapReduce's poor performance
- The chosen tests did not use the niche that MapReduce fills
    - Good: Parsing large datasets
    - Bad: Joining, Aggregation

Comparison

Parallel Database:

- Defined schema
- Has indexes
- uses declarative language (ex. SQL)

MapReduce:

- arbitrary format
- No built-in indexes
- Must create algorithms
- Considered the “brute force” approach

Stonebraker Talk

- Relational databases one size does not fit all (if anything)
    - Not going to work going forward
    - Column stores faster than row store
    - Complex analytics
        - Slow to simulate with SQL
        - Need statistics and data management
    - Graph analytics
        - Simulate or use special order graph engin
- Huge diversity oriented toward specific markets

Advantages and Disadvantages

- MapReduce fits with Stonebraker’s ideas
- MapReduce fills a hole in the market where a traditional Relational Database would be insufficient
- MapReduce can be applied to a large variety of tasks that can benefit from being distributed
- MapReduce is easy enough anyone can learn it and start using distributed systems