The Google File System & A Comparison of Approaches to Large-Scale Data Analysis

Nicholas D. Bradford

10.20.16

Andrew Pavlo , Erik Paulson , Alexander Rasin , Daniel J. Abadi , David J. DeWitt , Samuel Madden , Michael Stonebraker, A comparison of approaches to large-scale data analysis, Proceedings of the 35th SIGMOD international conference on Management of data, June 29-July 02, 2009, Providence, Rhode Island, USA

Sanjay Ghemawat , Howard Gobioff , Shun-Tak Leung, The Google file system, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003, Bolton Landing, NY, USA

[Lance Fortnow]. (2015, June 20). *Michael Stonebraker IEEE ICDE 2015 Ten-Year Influential Talk*. [Video File]. Retrieved from https://youtu.be/9K0SWs1mOD0

The Google File System – Main Idea

- Developers at Google recognized that their data storage system was no longer satisfying their growing needs as a company.
- Google File System was designed and implemented to be a scalable file system.
- The system was intended to be running on inexpensive hardware and the delivery of high aggregate performance.
- GFS was designed to work using a sprawling system of a thousand smaller, faster processing servers that could be repaired and rerouted with ease.
- GFS was also made to work to handle larger files, append data instead of rewriting it, and allow for simultaneous access of various data and programs.

The Google File System - Implementation

- The GFS is physically designed using inexpensive commodity components that commonly fail and must be replaced or waited on to be repaired.
- GFS stores larger than traditional files ex. Multi-GB files
- GFS handles two types of access traffic, large streaming read which includes reading MBs and small random reads that read small KB bits of data.
- Workloads that support large writes and appending data files
- Allow for multiple clients to simultaneously append data
- Bandwidth > Latency

The Google File System - Analysis

- A useful, cheap, and maneuverable system that fulfills Google’s need
- A more powerful system than previous versions due to ability to store and access larger bulk files
- Design concepts that allow for the appending of data and therefore less processing time when updating the system
- Overall: **B+** System due to potential faults of using less reliable but cheaper machines

A Comparison of Approaches to Large-Scale Data Analysis – Main Idea

- A group of professors and researchers from various Universities and Companies ran experiments to evaluate both the MapReduce and the Database Management System paradigms of data-analysis.

A Comparison of Approaches to Large-Scale Data Analysis – Analysis

- The study found that while the MR has perks in the ease of setup and low pricing, DBMSs are overall more consistent and useful.
- DBMS_X works at an efficient that is 3.2 times faster than MR and the Vertica System worked at an efficient that was 2.3 times faster than DBMS_X, showing the faster and potentially superior system.

A Comparison of Approaches to Large-Scale Data Analysis – Implementation

- The experiment was run on an open source Hadoop MR server and two parallel SQL DBMSs (Vertica and a major retail vendor)
- Benchmarks were used to establish a baseline test for both machines.
- Performance measured for parallelism against a cluster of 100 nodes

Google File System/A Comparison of Approaches to Large-Scale Data Analysis – Comparison

- GFS is a MapReduce system that works well for Google
- DBMS systems are more expensive and harder to implement but in the long run more successful
- While it is theorized that the GFS is superior to the Hadoop system tested in “A Comparison of Approaches…” the nature of a MR system is that it will be inferior to the DBMS.

Stonbraker Talk – Main Ideas

- There was fault in the DBMS theory, Model required polish
- Data Warehouse Market will soon we primarily Column Stores over Row Stores
- Despite this DBMS theory is a growth field and poised for innovation going forward

Advantages/Disadvantages (In Context)

Advantages

- Cheap Design
- Tailored to Needs
- Upgrade from former systems
- Quicker implementation
- Quicker appends

Disadvantages

- Less powerful in long run
- Chosen for cost over ability
- Slower System