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

http://www.labouseur.com/courses/db/papers/ghemawat.sosp03.gfs.pdf

http://www.labouseur.com/courses/db/papers/pavlo.sigmod09.comparison.pdf

Brandon Kline

10/29/17

The Google File System (GFS)

- The Google File System is able to run on inexpensive hardware while managing errors effectively and handling hundreds of requests at a time.
- GFS is able to minimize downtime through fast recovery and replica creation.
- GFS appends files rather than overwriting them as it is faster and wastes less space.
- GFS is also able to store a vast amount of data across a large number of machines with few issues.

The Google File System: Implementation

- Faults within GFS must be recognized and repaired instantaneously, as many files can be accessed concurrently by a large number of users.
- Each cluster contains a multiple “chunkservers”, which store file chunks, and a single “master”, which handles the related metadata.
- GFS creates snapshots of files that can be easily restored and replicas of master and chunk states in the event of a critical error.
- GFS can increase efficiency and minimize wasted space by detecting stale replicas and systematically collecting “garbage” data.

The Google File System: Analysis

- GFS is an exceptional file system, as it is able to store massive amounts of data in a compact and effective manner and swiftly correct any faults.
- The single “master” server and multiple “chunkserver” approach requires less stress on the master server, and allows for more rapid recovery should a chunkserver or the master server fail.
- The option to append files rather than overwrite them is also wise, as it is a much smaller workload than a full overwrite.

A Comparison of Approaches to Large-Scale Data Analysis

- This paper compares the performance of MapReduce to two parallel database management systems and explores the results with some benchmark tests.
- Relational databases and SQL have existed for over 20 years, while MapReduce is still fairly new.
- MapReduce is easier to use than other databases, as it only has two functions, but does not support relational schemas and lacks built-in indexes.
- The parallel databases support SQL and relational databases and are far more flexible and faster to execute than MapReduce.

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

- For the benchmark tests, the Hadoop system was used with MapReduce framework, while DMBS-X and Vertica were used as parallel DBMSs.
- The test itself involved scanning through a set of 100 byte records looking for a three-character pattern.
- Hadoop took some time before running at full capacity and took longer to reach the benchmark, but it held greater ease of use than Vertica and DMBS-X.
- By comparison, the parallel DMBSs were significantly faster, used less CPU power, but had less fault tolerance and were more difficult to use.

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

- While relational databases and SQL are relatively easy to learn compared to other languages such as C++, they could become impractical if the user is inexperienced or unfamiliar with databases.

- Hadoop and MapReduce may be far more simple than standard SQL, having just two functions, but they lack the flexibility and specificity that SQL does not.

- Ultimately, it depends on how familiar the user is with SQL and the scope of the query they desire. MapReduce is ideal for small simple queries, while SQL and relational databases are preferred for larger, more in-depth ones.

Comparison

- While the GFS focused more on the creation of the database and it's functions, the later paper explored already established databases and their ability to outperform one another.
- For it's implementation, GFS explained how the system could repair itself as well as how the system was structured, while the comparison paper discussed how functions in MapReduce and SQL were implemented while testing the benchmarks.
- One aspect that both papers touched on was failure models. Whereas GFS discussed the failsafes it had in place to prevent issues, the comparison paper compared the procedures Hadoop and DBMS-X/Vertica had and how they held up.

Michael Stonebraker on his 10-Year Most Influential Paper Award

- Relational databases from the 1970's to the 2000's were used in a broad, all encompassing approach to work as a "universal" database.
- By 2005, it was evident that streaming applications and column stores could not fit into a traditional database.
- Nowadays, individual markets have specific database applications, none of which row-store databases are good at mimicking.
- Given time and more advances in technology, it's likely that relational databases will become entirely obsolete in the near future.

Advantages and Disadvantages of the Google File System

- The main advantage of the Google File System is it's efficiency. It is able to store large quantities of data while being resistant to errors and crashes.
- However, a notable disadvantage is that, unlike MapReduce, the system is not the most user friendly, being similar to SQL in difficulty.
- Though, it should be noted, it is unlikely that this system will be used outside of Google on a mass scale in any capacity, so difficulty is less of a concern.
- In the context of the Stonebraker talk, GFS is a perfect example of a specific database for a niche market and the shift away from relational databases.