“The Google File System”

Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung

“A Comparison of Approaches to Large-Scale Data Analysis”

Pavlo, Andrew, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. Dewitt, Samuel Madden, and Michael Stonebraker

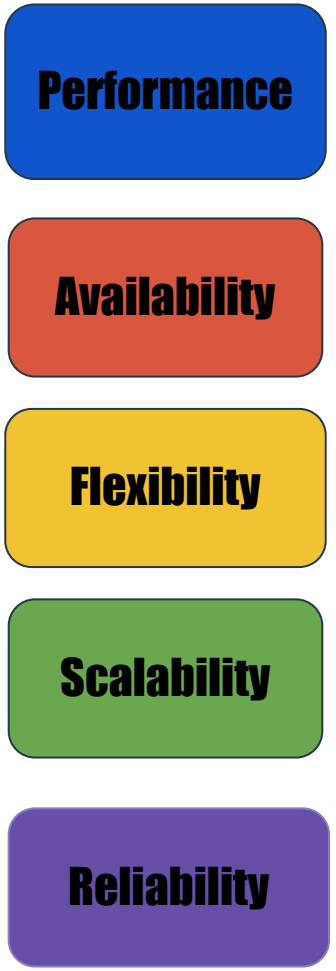
Katerina Tzannes

October 30, 2017

The Google File System(GFS):

"... a scalable distributed file system for large data intensive applications"

- Supports large-scale data processing and many clients
- Handles both, large streaming reads and small random reads
- Secure and fault-tolerant
- Failure is a norm rather an exception
- Scans itself to detect, tolerate, and recover from failures
- Prioritizes high sustained bandwidth than low latency
- Checksums check data integrity and detect corrupt data
- Clients interact with master to receive locations of chunk servers with necessary data



Implementation of GFS

- Single master with multiple chunkservers increases simplicity
- Master stores metadata(3 types) of files and chunkservers
  (Exception - location of a chunks replica is received at startup)
- Chunks are files broken down and can be up to 64MB
  (larger file capacity decreases the masters workload)
- Deleted files go to garbage collection for 3 days, then the storage is freed
  (unless deleted twice intentionally, it is not immediately removed)
- Shadow masters provide read-only access, enhance readability of files not mutated
  (updated by operation log and is available even when the master is down)
- Record appends preserves atomicity (no new files created, appends data)

Analysis of the Google File System

The GFS has achieved all its goals and is a good system, however (like everything) it is also flawed because it is specific to Google’s process and workflows.

Comparison Paper

- Compares two paradigms of large scale data analysis by performance, development complexity, and ease of use, while explaining its strengths and weaknesses and implementation
  - MapReduce (MR)
    - Relatively new
    - Better fault tolerance
    - “Schema Later” or “Schema Never” paradigm
  - Parallel Database Management System (DBMS)
    - Almost twice as fast, with better performance
    - Flexibility of joining multiple tables

Implementation

MapReduce (Hadoop)

- Better failure model
  (when a node fails another is used)
- Does not have the capability to execute the join task
- Considered “cold” because it has a slow startup until all the nodes can run at full capacity
- Lacks structure
- Written in Java

DBMS (DBMS-X, Vertica)

- The entire query must be restarted in the case of a node failure
- Excellent performance outcome of using indexes for selection tasks and benefit of a table for aggregation tasks

Analysis

Both DBMS-X and Vertica outperformed Hadoop when tested using the grep task and analytical tasks. Parallel DBMS however is more expensive to use and maintain.

MapReduce and DBMS are ultimately moving toward each other and combining aspects of both systems will most likely create an even better system.

Comparison of the Two Papers

- Share a relation to big data processing and can handle large amounts of data and storage
- Implementations of SQL and schemas with a set program could potentially improve the GFS
- The Hadoop Distributed File System(HDFS) could be replaced by the GFS

One Size Fits All - An Idea Whose Time Has Come and Gone

- Original mentality in the 80's and 90's believed RDBMS is always the answer and should be "universal"
- One size does not fit all, and one size fits none
- Although row stores may be faster by two orders of magnitude, they are not the best for streaming, transactions, etc.
- Transaction processing (OLTP) moving towards main memory deployments
- Currently huge diversity of engines oriented towards a specific application

Its a great time to be a DBMS researcher!

Advantages and Disadvantages of the GFS

Advantages:

- Movement from
- GFS is a distributed file system and will be needed for reliable storage and multi-user access
- Although GFS follows the MR model and relates to most of the markets Stonebreaker spoke about since it was specifically tailored for its goals it may not suffer the same pitfalls as a typical MR model

Disadvantages:

- Currently specialized and modeled to specifically Google and must be adjusted in order to be compatible with the workflow of other company's
- Not optimized for small files
- Does not have a schema and does not implement the relational rules
- When compared to a DBMS the performance may not be as good