A Comparison of Big Data Database Systems

By Nathan Fahrner
12/07/2014

Comparison Research Papers:

"**The Hive: A Petabyte Scale Data Warehouse Program Using HADOOP",**
Facebook Data Infrastructure Team: Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu and Raghotham Murth

"A Comparison Of Approaches To Large Scale Data Analysis" ,Andrew Pavlo Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, Michael Stonebraker

The Hive

Hive was developed to solve the problem of complicated coding of simple queries on the Map-reduce database system HADOOP. Queries had to be coded by hand, and could take days. Also, their was no standard way to transfer type definitions between developers. Facebook encountered this problem, and started to develop the Hive, an open source library that runs on-top of HADOOP. It simplifies queries on a HADOOP distributed file system with SQL like syntax. It also adds a system catalog, which can actually increase performance, and allows better query optimization. Hive provides structure to data, in tables, partitions, columns and rows. Schema can be described in queries, and stored in system catalog. Hive also provides easier interoperability with different mapping or translating functions from other languages. Its implementation of Map-reduce is very efficient, and makes it much easier for developers to write and share queries.

Implementation of the Hive

The hive runs as a driver on-top of HADOOP, in conjunction with a system catalog, a communication server, and interfaces. The driver, besides managing sessions, has three parts: a compiler, an optimizer, and an executer. The coiler and optimizer translate your SQL like statement into a set of aggregated HADOOP map/reduce jobs. It uses the system catalog to optimize these queries, and retrieve all schema/type data. The executor sends these jobs to HADOOP. How data is stored logically is controlled by the HADOOP driver. The system catalog is very useful for increase efficiency of queries. Meta-data about tables, partitions, columns, types, etc is also stored, allowing optimization of frequent queries. The communication server allows other applications to interact with the Hive driver using JDBC and ODBC interfaces. There are other interfaces for communication with the Hive driver, such as the command line and web interfaces. It also include extensibility interfaces that allow users to create their own functions, UDP and UDAP. It also uses the SerDe and ObjectInspector interfaces to handle type serialization and description.

My View Of The Hive

I believe the Hive is a great idea. It makes Map/Reduce database systems act much more like a relational database. Simplifying the syntax to SQL like statements brings makes this type of DBMS more accessible to users, and more portable. The addition of a system catalog to store types, schema, and meta-data not only increases performance, but makes relations and types clearer. It also allows one to monitor and optimize performance. Manipulating data as tables, columns, rows, etc makes it easier to use for those familiar with RDBMS. The HIVE can make it much easier to reuse code, and for developers to work together. Lastly, its open source so its accessible and modifiable to all.

My View of Parallel RDBMS vs Map-reduce

This article was very surprising to me. I had believed that RDBMS would be to slow for large clustered databases. The results of these tests showed in many cases this is not true. The results showed Map-reduce was slower because of a few factors: the startup of the workers nodes, the communication traffic between nodes, initially having to scan the entire record in the mapping function, and file transfer collisions. Many of these problems could be fixed with a different implementation.The parallel relational database systems took much longer to load data, as it had to create an index, but because of this the initial query executes quicker. They also have less communication and data transfers between nodes, which slow the entire process. Map-reduce had much better fault tolerance, only having to restart a workers process, while parallel DBMS had to restart entire queries. I think that Map-reduce would be better for systems with constant streams of data though do to the loading process of parallel DBMS. Map-reduce seemed to be easier to install as well. Its just got to be very hard to constantly code new queries. I also thought it was interesting to see that Vertigo stores data in columns, and its performance was good.

The Hive with Reflection of Comparison Paper

Hive has many advantages. Its easy to write queries, and makes use similar to a parallel RDBMS. The addition of a system catalog allows optimization of queries, and may speed up some of the processes that slow down Map-reduce databases. Its just as easy to transport types and queries as the RDBMS to other developers. It creates easy interoperability with external software. Using well known structures as tables,columns and rows makes this easy to understand to RDBMS users. A map reduce system seems like it would be quicker with data constantly being added, as it has a quicker loading process. Hive is very fault resistant.

Hive also has many disadvantages. The performance issues in the article will still apply to HADOOP as its a Map-reduce system, though its a different implementation. Relations are not enforced within the system, so users can enter bogus data, though the use of a system catalog reduces this problem. Data can't be inserted in a table, a new table must be created every time. Parallel RDBMS seem to run quicker than Map-reduce on fewer nodes.

Overall, Hive has its uses. It seems to be a better solution for Facebook, with constant streams of data. Map Reduce could still be improved. Still, parallel RDBMS seems to perform better, and enforces safety of data. With improvement in code, parallel RDMS could be a better solution for massive big data bases.