HIVE – A PETABYTE SCALE DATA WAREHOUSE USING HADOOP

ASHISH THUSOO, JOYDEEP SEN SARMA, NAMIT JAIN, ZHENG SHAO, PRASAD CHAKKA, NING ZHANG, SURESH ANTONY, HAO LIU AND RAGHOTHAM MURTHY

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

FAITH MATTHEW

3/2/17

HIVE: MAIN IDEAS

Hive is an open source petabyte scale data warehousing solution built on top of Hadoop, a MapReduce implementation. Hive was inspired by Hadoop's shortcomings

Hive structures data into tables, rows, columns and partitions and compromises a subset of SQL

Users can implement their own file formats and extend the system with their own functions and types

Data Model, type system and HiveQL slightly differs from traditional databases.

HIVE: IMPLEMENTATION

- Meta Store- The component that stores metadata about tables and partitions and acts like a System Catalog for Hive
- Driver- This component maintains sessions and manages the life cycle of a HiveQL statement
- Query Compiler- Metadata stored in Meta Store is used by query compiler to generate execution plan
    - Parse- Hive uses Antlr to generate AST (Abstract Syntax Tree) for query
    - Type Checking and Semantic Analysis- compiler retrieves information on all inputs/outputs to generate logical plan. Catches any possible compile/semantic errors. Transforms nested queries into parent-child relationships
    - Optimization The optimization logic consists of a chain of transformations such that the operator DAG resulting from one transformation is passed as input to the next transformation.
- Execution Engine- Executes the tasks produced by the compiler in order. This engine interacts with the underlying Hadoop
- Hive Server- Provides way of integrating Hive with other applications, a thrift interface, and a Java Database Connectivity/Open Database Connectivity
- Client Components- Command Line Interface, web UI, JDBC/ODBC driver, etc.

HIVE: ANALYSIS OF IDEAS AND IMPLEMENTATION

- Considering the fact that Hive is a work in progress, I believe it's off to a great start
- However it has numerous limitations that need to be addressed such as not being able to insert data into an existing table or data partition. All inserts overwrite existing data.
- It's very easy to use and get acquainted with. Both beginners and advanced users
- It's scalable and can be used for both small and large projects for a low price compared to other infrastructures

COMPARISON PAPER MAIN IDEAS

- Compares and analyzes MapReduce to Parallel Database Systems and their respective approaches to performing large scale data analysis
- MapReduce makes use of only 2 functions which are Map and Reduce. Function is written by user to process key/value pairings
- Parallel Database Systems are capable of running on clusters of nodes, support standard relational tables and SQL
- Parallel Database Systems seem to have a large advantage over MapReduce for large scale work and projects
- However Hadoop was much easier to set up and use. Plus, it has extensibility which the parallel DBMS lacked

COMPARISON PAPER IMPLEMENTATION

- MapReduce and Parallel DBMS completed 4 tasks. How well and efficiently these systems completed the task were recorded
  - Selection Task- this task is a filter to find pageURLs in the rankings table with a page rank above a user defined threshold
    - Parallel DBMS outperform MapReduce significantly in this task
  - Join Task- consists of two sub tasks. In the first task each system must find the sourceIP that generated the most revenue within a particular date range. In the second task the system must then calculate the average pageRank of all the pages visited during this interval
    - Big performance difference between MapReduce and the parallel database systems
  - Aggregation Task- calculate the total adRevenue generated for each sourceIP in the UserVisits table, grouped by the sourceIP column.
    - The two parallel Database systems once again out perform Hadoop
  - UDF Aggregation Task- The final task is to compute the inlink count for each document in the dataset, a task that is often used as a component of PageRank calculations. Specifically, for this task, the systems must read each document file and search for all the URLs that appear in the contents.
    - Parallel DBMS performs worse than Hadoop due to the added overhead of row-by-row interaction between the UDF and the input file stored outside of the database.

COMPARISON PAPER: ANALYSIS OF IDEAS AND IMPLEMENTATION

- There's a lot to learn from these respective systems and they both have a lot to offer.
- Despite the parallel DBMS outdoing Hadoop in most tasks, the MapReduce program had more extensibility, is more user friendly and minimizes work lost when a hardware failure occurs
- Both types of systems have their own on drawbacks and strengths. A user should pick a system based on what's best for them

HIVE VS COMPARISON PAPER

- Since Hive was inspired from Hadoop's shortcomings I think Hive is pretty powerful.
- Hive was able to increase the performance of Hadoop by 20% compared to Hadoop's performance for tasks in the Comparison paper
- Hive just seems like the go to for beginners
- Hive is an example of the integration of SQL and MR discussed in the comparison paper

STONEBRAKER TALK: MAIN IDEAS

- Relational Databases are now good for nothing. They are obsolete but used to work for everything
- Different markets will change over the years. Data Scientists will replace Business Analysts in Complex Analytics etc
- No better time to be a Database Researcher
- New technology will try to dominate market and fight over shares

DISADVANTAGES AND ADVANTAGES

- Hive already has limitations to begin with
- But Hive costs less compared to other systems
- Hive is great for both beginners and advanced users
- I believe Hive is still a good system despite Stonebraker claiming relational databases are obsolete