Big Data Paper Summary for CMPT 308:

By: Nick Carrozza

20 October 2016

Paper Titles:

"Hive − A Petabyte Scale Data Warehouse Using Hadoop"

"A Comparison Of Approaches to Large-Scale Data Analysis"

Paper References:

Pavlo, Andrew, Rik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, and Michael Stonebraker. "A Comparison of Approaches to Large-Scale Data Analysis." N.p., 2 July 2009. Web. 18 Oct. 2016.

Thusoo, Ashish, Sen Sarma Joydeep, Jain, Namit, Shao, Zheng, Chakka, Prasad, Zhang, Ning, Antony, Suresh, Liu, Hao, and Murthy, Raghotham. "Hive − A Petabyte Scale Data Warehouse Using Hadoop." N.p., 2010. Web. 18 Oct. 2016.

Hadoop and the Birth of Hive:

- Hadoop is a popular “open-source map-reduce implementation” used to store and process enormous amounts of data
- Hive was introduced as a solution to the difficulty of writing “low level” programs in Hadoop that were difficult to maintain and reuse
- Hive was built since Hadoop is difficult and inefficient compared to popular query languages like SQL
- This notion was very useful to companies like Facebook and Yahoo!, who often deal with a continuous stream of recorded data that becomes extremely costly and progressively difficult to manage

/* Main Goal of Hive = “bring data closer to users” */

How Hive Works:

- The team that built Hive (HiveQL) aspired to bring familiar SQL concepts to a very unstructured Hadoop
- The idea of making Hadoop more user-friendly is implemented by applying SQL-like application to the extremely large data sets in Hadoop
- Hive does this by structuring data according to relational database concepts that include utilizing tables, rows, and columns in addition to many similar data types
- Thus, the query language of HiveQL is very similar to SQL

Analysis/Implementation of Hive:

- It's evident that Hive's implementation is very similar to SQL, and the developers' intent was successful
- Hive helps save companies tremendous amounts of time by having the ability to incorporate data from different programs
- In the scope of database management, time is of interest to protect, arguably even more so than storage which is considered by some to be a more abundant resource
- Hives similarity to SQL is demonstrated with its query syntax (the use of SELECT, FROM, WHERE, GROUP BY, JOINS etc.)

SELECT Main_Ideas FROM Comparison_Paper;

- Purpose is to compare the use of MapReduce versus SQL DBMS for large scale data analysis
- Consider the difference between the systems in terms of indexing, programming models, data distribution, and query execution strategies as well as the subsequent trade-offs of using the two systems
- Conflict existed of whether to use relational or codasyl implementation in database systems, and a debate ensued over whether you should "state what you want" or present an algorithm to access data in a database

Implementation of the Comparison Through Testing:

- Numerous tests were conducted by the team to compare the differences of these two, and most tests concluded that SQL DBMS was overall superior to the MR model, evidenced with subjects like aggregations, selection task and data loading where SQL DBMS outperformed the MR model

- This was done with a series of benchmark tasks, where numerous measurements were taken including time to load test data, the “Grep task”, tasks related to HTML document processing, etc.

Results and Analysis of Tests:

	Pros	Cons
MapReduce	• Programmer has more control/autonomy over structuring data • Minimal work lost when hardware failure occurs	• Not good for long-term/larger projects • No schema so each user must write a custom parser; more complicated for sharing information
SQL DBMS	• Use of data sharing via the system catalog • Much faster than MR • Better execution strategies • Better for aggregations	• Non-procedural, many used to traditional programming languages will find it challenging

/* These pros and cons are consistent with the team's conclusion that SQL DBMS is overall superior to the MR model */

```
SELECT Ideas, Implemetations
        FROM Hive_Paper
        RIGHT OUTER JOIN Comparison_Paper ON Hive_Paper.Ideas =
                                        Comparison_Paper.Ideas
```

Hive

- Created in an effort to simplify the accessing of data in Hadoop
- Hive was an attempt to establish a SQL-like method of retrieving data from databases
- The comparison paper is consistent with the rationale of creating Hive

Comparison paper

- Basic conclusion was that SQL DBMS is superior to the RM model for various reasons
- Tests involving multiple different aspects of the database, including aggregations, selection task and data loading suggest this idea

/* Both papers conclude that relational database systems are the preferred type of database system. */

SELECT Main_Ideas FROM Stonebreaker_Video

- According to Stonebreaker, a huge diversity of engines exist for which traditional row stores are considered “good for nothing”
- Where RDBMS used to be “the answer”, that is no longer the case and new implementations will come about in the future as a result of new ideas
- Relational Database systems were once viewed as a “one size fits all” implementation, and Stonebreaker elaborates his belief that “one size fits *none*”
- This belief was confirmed by 2015 and was questioned by Stonebreaker beginning in 2005

How Useful is Hive?

Advantages of Hive

- Allows easier usage of complicated Hadoop using traditional SQL implementation
- Comparison paper suggests SQL DBMS is preferable to other types of database management systems including the MR model, rendering Hive very useful

Disadvantages of Hive

- According to Stonebreaker, SQL-like implementation will become obsolete in the future as new ideas are implemented
- This suggests Hive, while currently useful, will no longer be relevant in the future as new implementations are created