Big Data Paper Summary

By: Carolena Realmuto

PAPERS ANALYZED:

HIVE-A PETABYTE SCALE DATA WAREHOUSE USING HADOOP

THUSOO, ASHISH , JOYDEEP SEN SARMA, NAMIT JAIN, ZHENG SHAO, PRASAD CHAKKA, NING ZHANG, SURESH ANTOMY, HAO LIU, RAGHOTHAM MURTHY. HIVE – A PETABYTE SCALE DATA WAREHOUSE USING HADOOP. FACEBOOK DATA INFRASTRUCTURE TEAM. IEEE. 2010

A COMPARISON OF APPROACHES TO LARGE SCALE DATA ANALYSIS

PAVLO, ANDREW, ERIK PAULSON, ALEXANDER RASIN, DANIEL ABADI, DAVID DEWITT, SAMUEL MADDEN, AND MICHAEL STONEBRAKER. A COMPARISON OF APPROACHES TO LARGE-SCALE DATA ANALYSIS. SIGMOD. JULY 2, 2009.

Hive- The Main Idea

- Hadoop, a popular open-source map-reduce implementation, was originally used by Facebook but was more harmful than useful.
- Facebook needed a program that used popular query languages, was time efficient, easy to use by end-users, while being capable of scaling large data sets.
- They created a solution in 2007 called Hive. It is an open-source data warehousing solution built on top of Hadoop that structures data into tables, columns, rows and partitions.

Hive- Idea Implementation

- It has a query language similar to SQL called HiveQL.
- Each column has an associative type – primary (integer, string, floating) and complex (arrays, lists, structs).
- It does not impose any file format restrictions (it is specified when table is created).
- The system catalog for Hive is called Metastore - stores all the information about the tables, their partitions, the schemas, the columns and their types, the table locations etc.
- Tasks are executed in the order of their dependencies.

Hive- Analysis

- Hive has enabled the huge ad hoc query workload on the Hadoop cluster in Facebook because of the simplicity with which ad hoc analysis can be done.
  - Potential problem: ad hoc jobs are unpredictable and may not always be properly executed so valuable cluster resources are consumed.
    - This leads to degraded performance of reporting queries which are time critical.
- Hive enabled Facebook to provide data processing services to engineers and analysts at a fraction of the cost of a more traditional warehousing infrastructure.

Comparison Paper – The Main Idea

- "Cluster computing" harnesses large numbers of (low-end) processors working in parallel to solve a computing problem.
- There are two approaches to large scale data analysis: MapReduce vs. Parallel SQL
- MapReduce:
  - It consists of two functions, called Map and Reduce, that are written by a user to process key/value data pairs.
  - MR permits data to be in any arbitrary format.
- Parallel SQL:
  - DBMSs require that data conform to a well-defined schema.
  - Most (or even all) tables are partitioned over the nodes in a cluster.
  - It uses an optimizer that translates SQL commands into a query.

Comparison Paper – Idea implementation

- MapReduce:
  - Is well suited for development environments with a small number of programmers and a limited application domain.
  - Does not provide built-in indexes – programmer must implement their own (not easy).
  - Provides a more sophisticated failure model - if a unit of work fails, then the MR scheduler can automatically restart the task on an alternate node.
  - Benchmark used: Hadoop
- Parallel SQL:
  - Requires data to fit into the relational paradigm of rows and columns
  - Uses hash or B-tree indexes to accelerate access to data
  - Uses a push approach to transfer data instead of a pull
  - Benchmarks used: DBMS-X, Vertica

Comparison Paper – Analysis

- Hadoop:
  - MR is always forced to start a query with a scan of the entire input file
  - Set up was easier than other two databases
  - Lack of schema - each user must write a custom parser, complicating sharing data among multiple applications; parsing records at run time is inevitable
  - Higher level interfaces (like Hive) are being used in conjunction with Hadoop to make it more efficient and simplify complex coding
- DBMS-X/Vertica:
  - performance advantage over Hadoop MR in executing a variety of data intensive analysis benchmarks - DBMS-X was 3.2 times faster than MR & Vertica was 2.3 times faster than DBMS-X
  - In Vertica, only those columns that are needed to execute a query are actually read from disk
    - This reduces the amount of disk I/O that is performed to execute a query
  - Lacked extensibility
  - SQL is a powerful tool
- Final verdict: Parallel databases have a larger long-run potential than MR

Michael Stonebraker – Main Ideas

- In the past, "RDMS is the answer"…
- However in the present, "one size does not fit all."
  - Traditional row stores (DB2 , SQL server, oracle) are obsolete; aka "good for nothing" and products will have no market share in the future
  - Markets such as Data Warehouses, OLTP, NoSQL, Analytics, streaming, and graph databases will have column stores in the future
    - Makes systems faster and more efficient
  - We will see new implementations within NVRAM, Vectorization, big main memory, etc
- With transition from old to new, vendors will have a hard time morphing without losing market share
  - He believes SQL server and Hekaton will lead the way
  - Expects systems to maintain common user interface and replace old engines
  - "Main tent" conflict → SAP will make product not available for ORACLE
- In 80s and 90s, DBMS was "dead on our feet." This could be blamed on "one size fits all" philosophy which Stonebreaker says is completely dead.

Hive – Advantages and Disadvantages

- Advantages:
  - MR has easier set up
  - May be better than RDBMS for simple or one-time processing tasks
  - Useful for large datasets to query
  - Extensibility friendly- has a range of user function APIs that can be used to build custom behavior in to the query engine
  - Easily scalable at low cost
- Disadvantages:
  - Lack of schema prevents the performance improvements enabled by B-trees and has partitioning
  - Stonebraker states it's "too low level"
  - RDBMS is more efficient with complex processing and when data is used across an enterprise
  - Hive lacks an update/delete function. Data can only be selected and added.
  - Slower