Building a High-Level Dataflow System on top of Map-Reduce: The Pig Experience

Alan F. Gates, Olga Natkovich, Shubham Chopra, Pradeep Kamath, Shravan M. Narayanamurthy, Christopher Olston, Benjamin Reed, Santhosh Srinivasan, Utkarsh Srivastava

—

Summary by Brandon Litwin 3/7/17

Pig: Main Idea

- Created as an improvement to Hadoop's MapReduce system
- More efficiently manipulates large pieces of data
- MapReduce is too simple to handle really complex data
- MapReduce requires users to code dataflows by hand, allowing a lot of user error
- Pig was challenging to develop but is being praised for accomplishing what it set out to do

Pig: How it is Implemented

- Compiles dataflow programs written in Pig Latin programming language into MapReduce jobs
- Executes them through Hadoop's system
- Pig's commands can be executed in Interactive, Batch, and Embedded modes
- Pig uses storage functions to delimit data values when loading data
- Has its own data types: Map, Tuple (ordered list of data), and Bag (collection of tuples)
- Unknown data types are given type bytearray
- Translates logical plan to physical plan to be put through MapReduce

Pig: Analysis and Interpretation

- Pig saves time by creating a logical plan before putting data in MapReduce
- Easier customization and management of files by the user
- Pig commands are easier to understand by the user than plain MapReduce code
- Handles unknown data types in a logical way through bytearray

Comparison Paper: Main Ideas

- MapReduce is a tool for “cluster computing” that competes with the parallel DBMS of the past (SQL and Oracle)
- Compares the way data is implemented in both types of systems
- Compares the architectural elements of the systems
- Compares the performances of the systems

Comparison Paper: How it is Implemented

- In MR, Map reads the input file and filters it, and Reduce combines the records into an output file
- SQL is much faster to execute but takes longer to load the data beforehand
- MR's lack of constraints make it impractical for large projects
- MR's simplicity requires programmers to write their own indexes, which is prone to mistakes
- MR is more flexible, but open source projects like Ruby on Rails have increased SQL's flexibility while maintaining SQL's ease of use
- Detailed performance comparisons show that parallel DBMS outperform Hadoop

Comparison Paper: Analysis and Interpretation

- MapReduce is a good idea that is not yet perfected
- It’s surprising that traditional DBMS outperform MapReduce
- This paper asks if MapReduce is truly a better replacement of old systems
- Luckily, there are community efforts to improve MapReduce

Comparing the Papers

- Both papers recognize the disadvantages of the MapReduce system
- Pig's goal is to use their own system with MapReduce to improve the functionality of MapReduce
- Comparison paper mentions Pig as part of the MR community that is working to improve its functionality
- Pig paper is possibly more biased towards mentioning MR's weaknesses than the comparison paper
- Pig paper wants to show off how what they created is useful in solving MR's weaknesses

Michael Stonebraker's Talk

- Relational DBMS were hailed as being the universal solution
- He argues that now these systems are completely irrelevant to today's needs
- Column stores are a better alternative to row stores
- SQL was good for business analysts who wanted basic data
- Number of data scientists are growing, and SQL analytics are not complex enough for them
- Column stores will become more useful for these changing markets

Pig and MapReduce in the Context of Stonebraker's Talk

- Pig and MapReduce are improvements of Hadoop's system of manipulating big data
- They are different from relational database systems
- However, they do not use column stores like Stonebraker would want
- Stonebraker believes MapReduce is a step in the wrong direction
- New systems need to be developed that use column stores