

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

&

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

By Kevin Mirsky

3/6/2017

The Pig Experience: Main ideas

- Map-Reduce has become a popular tool for processing large data sets
- However, it is very barebones; Lacks many common operations such as *join*
 - Thus, they must be coded by hand!
 - Higher chance of user created errors
 - Takes time to write and test
- Pig is a system that is built on top of Map-Reduce
 - Offers advanced processing tools and functions to make user's life easier
 - Uses high-level language similar to SQL for easier use
 - Desires to capture the best of both worlds



The Pig Experience: How did they do it?

- Pig programs are coded in a custom language called *Pig Latin*
- Pig Latin code is parsed and turned into a Logical Plan
- Logical Plan is then optimized
- Logical Plan is turned into physical plan, explicitly listing out all steps
- Physical Plan is then split up into Map-Reduce plan
- Plan is then turned into in jobs and sent to Map-Reduce program for execution



The Pig Experience: A Clever Idea

- Map-Reduce on its own seems difficult to use when doing advanced manipulation
- A tool to perform non-supported operations should:
 - Make operator's job easier
 - Reduce time to construct queries
 - Reduce opportunities for bugs
- Though it does add a performance overhead (1.5x of just Map-Reduce)
 - In my opinion worthwhile for the added abilities and ease of use

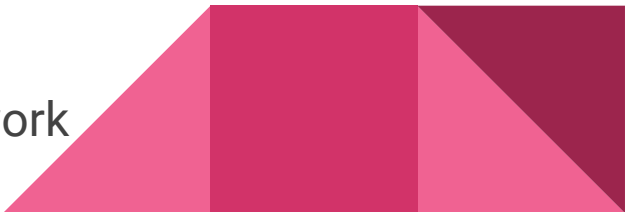
Overall, I think the authors of Pig developed a useful tool



A Comparison of Approaches: Summary

- Map-Reduce and parallel Database Management Systems (DBMS) are competing systems for analysis of data
- The authors wished to determine the strengths/weaknesses of both and whether one could be considered superior
- Authors assess usability as well as performance

Results:

- Found DBMS often had superior performance
 - Map-Reduce was far easier to set up
 - SQL code for DBMS required less work
 - Map-Reduce is more fault tolerant, minimizing lost work
- 

Comparison Implementation

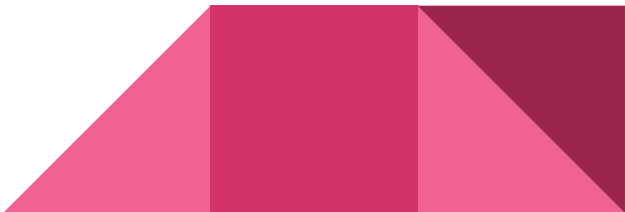
- The authors discussed elements of each tool which may impact performance or usability
- They then conducted several tests to measure speed of various tasks
 - Two DBMS systems were tested alongside Hadoop, an open-source Map-Reduce program
 - Also ran separate tests with varying number of processing nodes and workload sizes
- After receiving results, authors then determined what functionality created those outcomes
- Finally, they discussed how user-friendly/easy to use the systems were



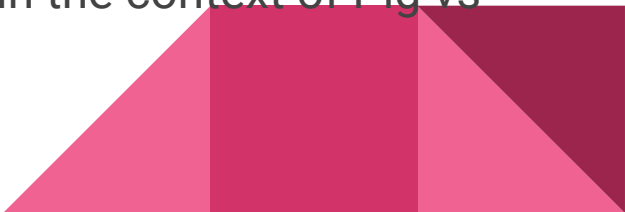
Comparison Analysis

- The method of benchmark testing seemed like the right way to test this
 - Speed is probably the most important metric to many
- Analysis of what behaviors influenced results was important
 - Others can use this to better make decisions on what to use
- Various types of tasks ensured tests weren't biased towards one tool's specialty
- Examining ease of use and usability was smart
 - Likely overlooked by many but important nonetheless

Based on the data, I'd conclude that DBMS systems are arguably superior



Comparison of the two papers

- Both assess usability of Map-Reduce
 - As assessed by the Comparison paper, Map-Reduce can be difficult to code compared to database systems
 - Required to write own algorithms in low level language
 - High level languages like SQL handle this for the user
 - Pig paper highlights this as well
 - Pig aims to remove this negative by creating a high level language for Map-Reduce
 - Both also assess performance, but Pig does only so in the context of Pig vs Ordinary Map-Reduce
- 

Stonebraker Talk

- People have been caught up with the idea that regular relational DB systems are the answer to everything
- In reality, there are far better implementations purpose-made for every market/task
- Specialized systems are superior to traditional general purpose systems
- The development of new database engines is where innovation is happening



Pros and Cons of Pig in Context

Pros:

- Makes Map-Reduce far easier to use by handling complex operations and simpler language (*Comparison Paper*)

Cons:

- Increases execution time of an engine that's already slower than DBMS systems (*Comparison Paper*)
 - No specialization (*Stonebraker & Comparison*)
 - No real strength other than simplicity
 - Lacks compelling reason to use when there's many other specialized options for your task
- 