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

By: 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

By: Daniel J. Abadi, David J. DeWitt, Samuel Madden, Michael Stonebraker

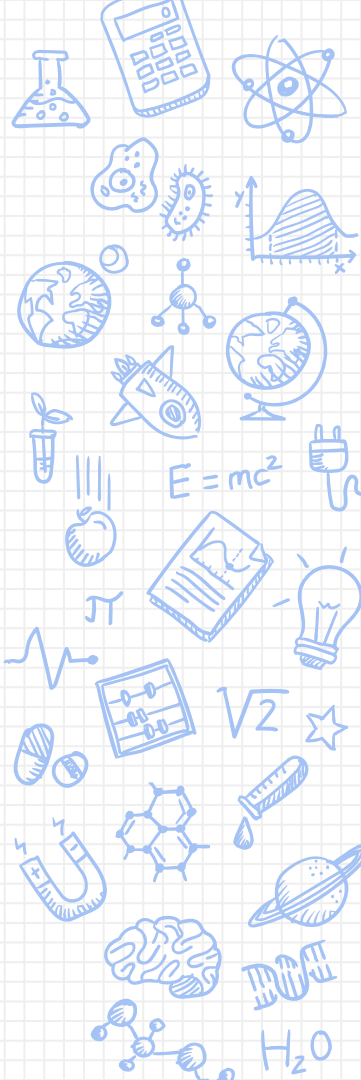
Big Data Paper Summary

Created by: Michael Gutierrez

March 6, 2017

Pig Main Ideas

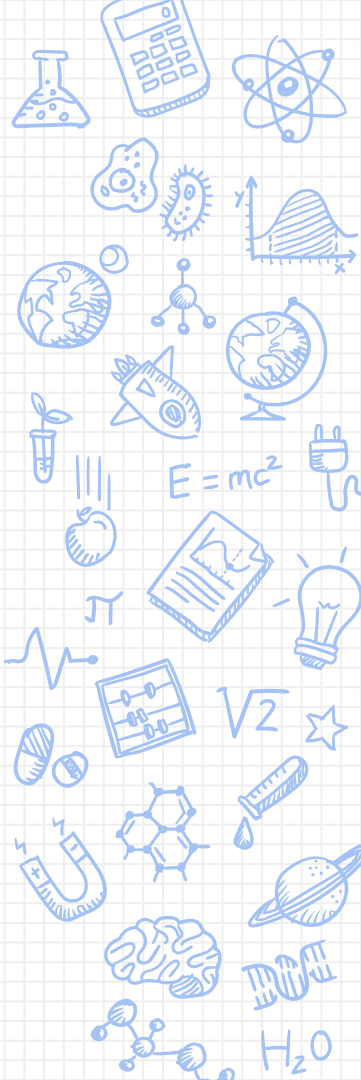
- Dataflow System that is a blend of SQL and Map-Reduce
  - Offers intelligent data manipulation
  - Also offers the Map & Reduce functions & executables
  - Dataflows interleave relational-style operations (such as joins) with user provided scripts
  - Can specify schemas for relational style operations
- Language for Pig is *Pig Latin*
  - A high level abstraction of Map-Reduce in the spirit of SQL
  - Can be extended with User Defined Functions written in different programming languages
- Compiles dataflow programs into sequences of Map-Reduce jobs in an Map-Reduce Environment on large sets of data
- It is open source!



Pig Implementation

The Pig System works in the following steps:

1. Takes Pig Latin program as input
    a. Supports standard scalar types (int, long, double,chararray) & three complex types (map, tuple, bag)
    b. Three Methods of Type Declaration: No type declaration, user-defined-function type declaration, Pig program type declaration
    c. Lazy Type Conversion: Conversion only until necessary
2. Compiles that program into one or more Map-Reduce Job(s)
    a. Logical Plan -> Physical Plan -> Map-Reduce Plan
        i. Pig Translates a logical plan into a a physical plan and places each physical operator inside a Map-Reduce stage to arrive at a Map-Reduce Plan
3. Executes those jobs on a Hadoop cluster
    a. Flow Control uses the push and pull model
    b. Pig does allow for limited nested programming
    c. Pig is implemented in Java therefore it has the same memory management issues

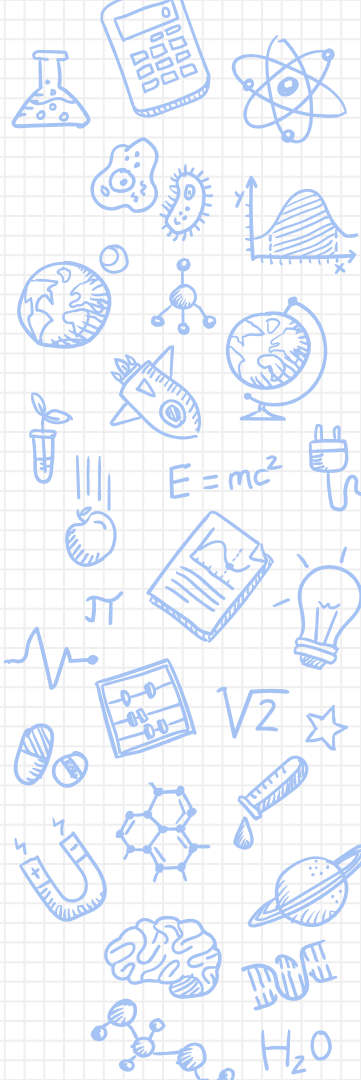


Pig Analysis

- Pig is an an interesting platform for creating Map-Reduce programs. It is essentially a high-level framework based on Hadoop in order to speedup development. It allows for its users to avoid the complexities of Hadoop by streamlining the process of Hadoop Map-Reduce job creations. This is a great way to entice programmers to adopt Pig as they do not need to know what is happening under the hood of Hadoop. This is achieved by incorporating the language elements of SQL so that Pig is easier to adopt as it has been a standard in the Database world for quite some time. The ability to create separate functions in Java is also another bone given to developers as it programming language that has been refined over time with great tools and support around it. Having Pig in open source community is also huge benefit as it can be contributed to by many different developers in order to keep it up to date and have community support.

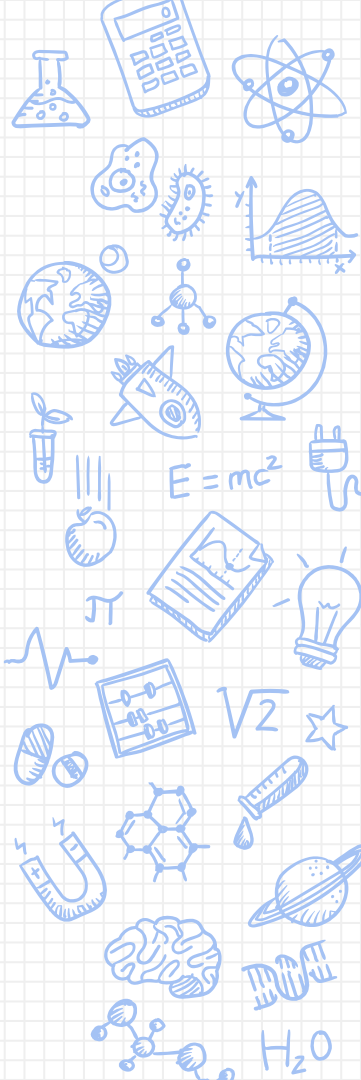
Comparison Paper Main Ideas

- A comparison paper that puts Map-Reduce against parallel SQL Database Management Systems for performance & complexity of development
- The “Cluster Computing” movement has risen
  - It is to utilize many low-end processors together to solve computing problems
- Map-Reduce
  - Popular because of its simplicity when expressing more sophisticated distributed programs
  - Native ability for data sets to be stored across partitions
  - Data can be arbitrary
- Database Management Systems
  - Has been available for two decades meaning there’s a lot of support
  - Supports SQL and relational tables
    - Makes Data more accessible for the users
  - Data must follow a well-defined schema
- Generally, SQL Database Management Systems were significantly faster & required less code to complete all the tasks



Comparison Paper Implementation

- Map-Reduce consists of 2 functions written by a user to process key-value data pairs
    - *Map*: Reads set of records from an input file, filters or transforms as desired, then a new set of records are outputted
    - *Reduce*: Input files are transferred over network from Map nodes' local disks and then combines the records to a final record on an output file
- Parallel DBMSs has two key aspects for enabling partitioning
    - All or most of tables are partitioned over nodes in a cluster
    - System uses an optimizer which translates SQL into a query plan
        - Execution is divided among multiple nodes
- Test designed to determine each system's: performance, fault tolerance and flexibility
    - Test for: Grep Task, Task Execution, Data Loading, Selection Task, Aggregation Task, Join Task, and UDF Aggregation Task
- Hadoop used as the contender for Map-Reduce
- Vertica & DBMS-X used as contenders for parallel DBMSs
- DBSM are concluded to beat Map-Reduce in most tasks

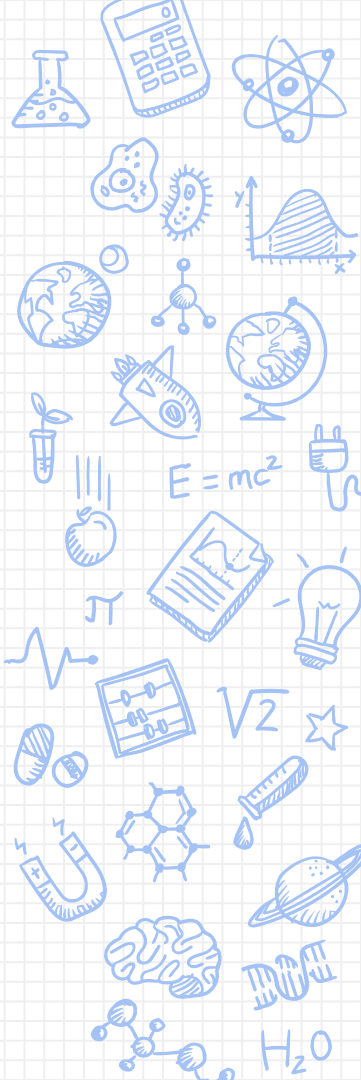


Comparison Paper Analysis

- Although Map-Reduce is a simpler system to set up it sacrifices in performance and flexibility, whereas parallel DBMSs require a little more knowledge and know-how to initialize it produces performance gains up three times faster than Map-Reduce.

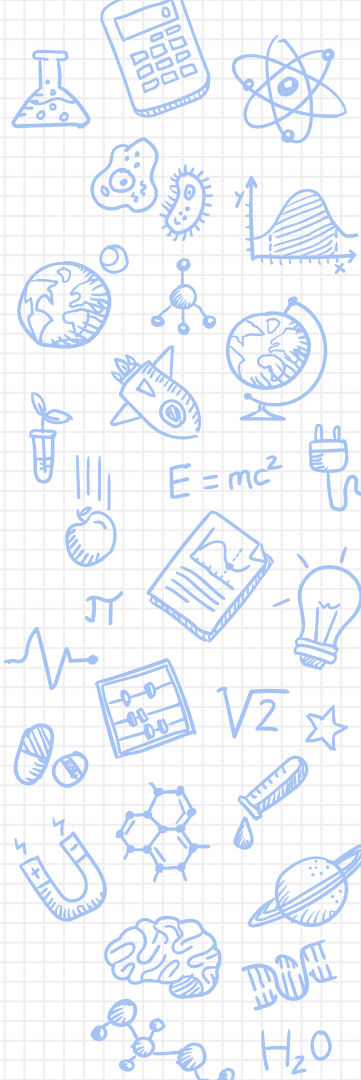
Pig vs Comparison Paper

- The two papers are somewhat different from one another
- The Pig paper focused more on the benefits and improvements of a framework built upon Hadoop
- The Comparison paper focused on benchmarking two systems head-to-head when handling different kinds of tasks: Map-Reduce vs parallel DBMSs
    - Concluded that you should probably be using DBMSs instead of Hadoop's Map-Reduce
- The papers are similar in that they both detail benefits of using other systems in comparison to Hadoop
    - There are benefits to using a framework built ontop of Hadoop or using an entirely different kind of system all together such as Vertica



10 Year Test of Time Main Ideas

- As much as we want there to be, there is no data model that fits all
- The Relational Database Model introduced abstract data types, referential integrity, etc.
- The Relational Database Model has been widely adopted and has been attempted to be used as the solution for all of of our problems however it cannot achieve this feat
- In fact, one size fits none!
    - *Data Warehouse Market* will move toward column stores as they are much faster
    - Complex Analytics are using array database systems as SQL is not effecient for these tasks
    - Streaming Market are widely using OLTP engines
    - Graph Analytics simulate in a column store or in an array engine as row stores are not very effective
    - OLTP is moving toward main memory deployment and row stores are resource hogs
    - NoSQL Market has expanded dramatically with many data models with no set standard
- The 'Titans' of the industry will respond by using many different engines underneath a common parser and maintaining their user interface
- Now there are a diversity of solutions to our problems now in the 21st Century



Pig and Stonebreaker

Advantages

- Pig is an open source framework that is built upon the Map-Reduce System. This means that it will constantly be improved upon in order to keep it with up technological demands of the future
- Pig is a blend of both the old and new: SQL and Map-Reduce

Disadvantages

- It will not be able to solve all of the world's computing problems
    - Might have problems that we do not know of now that will make Pig not as ideal of a solution
- The authors may have compared Pig to vanilla Map-Reduce however they have not benchmarked Pig versus the popular DBMSs