MapReduce, A Comparison of Approaches to Large-Scale Data Analysis, Stonebraker video

RYAN OWENS

OCTOBER 20, 2016

Main Idea of MapReduce

- MapReduce is split into two separate functions: Map & Reduce
- The Map function is specified by the user, processing a key/value pair which generates a set of intermediate key value pairs
- The Reduce function takes the set of intermediate key/value pairs and combines the values returned by the map function into a readable format.
- The map function may be parallelized, which splits the processing among a large cluster of multiple machines which reduces the processing time needed to map specified data. Such as the frequency of words.

MapReduce Implementations

- The implementations for MapReduce involve large clusters of machines which divide the data to process in parallel with each other.
- Each machine is assigned to either the Map task, or the Reduce task depending on necessity.
- Once there are key/value pairs from the Map function, they are then passed to the Reduce function grouping all values of the same key together.

Analysis of MapReduce and it's implementations

- MapReduce is a great way to analyze big data by going through files and mapping key/value occurrences, and reducing the values which give us a frequency of use.
- In utilizing large clusters of machines, MapReduce is able to parallel process massive quantities of data fairly quickly which in turn gives information through identifying patterns within large-scale data.

Main Ideas of A Comparison of Approaches to Large-Scale Data Analysis

- The main idea of this paper is comparing MapReduce and parallel DBMS's and understanding the differences between them.
- Informing readers that both approaches can return can achieve desired outcome using either specified queries through a parallel DBMS or utilizing MapReduce jobs.

A Comparison of Approaches to Large-Scale Data Analysis Implementations

- There is a trade-off of execution time and time to load data
- MapReduce is much faster with loading data and getting the general set up ready.
- DBMSs are much faster with execution but take longer to set up and load the data
- Both are able to perform similar functionality, parallel computations by partitioning data and using a large cluster of machines, or nodes, to traverse through the data partitions and ultimately filter the data.

Analysis of A Comparison of Approaches to Large-Scale Data Analysis' idea and its implementations

- MapReduce is much simpler than the DBMS approach, having only two functions – Map and Reduce
- DBMS can be difficult to use because the queries can be more complex to set up
- DBMS parallelization requires data to conform to a table schema of rows and columns.
- MapReduce requires more time than a DBMS to get all nodes to reach and run at max efficiency.

Main idea of Stonebraker video

- Potential implementations for RDBMSs based on current practice comparisons, predicting where RDBMSs markets are heading.
- Tried to create a one size fits all RDBMS using row stores
- Row stores are bad with handling many big markets
- Column stores are two orders of magnitude faster than row stores.
- “The Innovators Dilemma,” – When innovation takes hold, vendors of older version have a difficult time changing to the new version.

Advantages/Disadvantages of MapReduce in context of the A Comparison of Approaches to Large-Scale Data Analysis and Stonebraker's video

- MapReduce will work with data in any format
- Can recover from faults in the middle of executing a query
- MapReduce is quick and easy to set up
- Slower execution time compared to parallel DBMSs
- MapReduce is great with storing work when hardware fails
- Simple to use having only two functions – Map & Reduce