BIG DATA PAPER SUMMARY

- MAPREDUCE: SIMPLIFIED DATA PROCESSING ON LARGE CLUSTERS (JEFFREY DEAN AND SANJAY GHEMAWAT)
- A COMPARISON OF APPROACHES TO LARGE-SCALE DATA ANALYSIS (ANDREW PAVLO, ERIK PAULSON, ALEXANDER RASIN, DANIEL J. ABADI, DAVID J. DEWITT, SAMUEL MADDEN, AND MICHAEL STONEBRAKER)
- MICHAEL STONEBRAKER ON HIS 10-YEAR MOST INFLUENTIAL PAPER AWARD AT ICDE 2015 (MICHAEL STONEBRAKER)

BY CLAYTON SZELESTEY, 3/7/17

MAPREDUCE: MAIN IDEA

- Make large scale data processing easier to implement/use
  - Make parallelization easier for user
  - Make fault tolerance easier for user
  - Make load distribution simpler for users
- Make writing queries for data simple for users
- Reduce bandwidth cost
- Increase efficiency of basic MapReduce model
- User customizability
- Make use of large clusters of machines

MAPREDUCE: IMPLEMENTATION

- Hide details of parallelization, fault tolerance, and load distribution from user through layers of abstraction
- Queries are made of two parts (Map and Reduce) that are written by users. Map takes input and creates an intermediate set of key/value pairs which the reduce function uses to create the final output
- Alternate version of MapReduce is available for a single machine for use in debugging
- Master computer tells workers to read input data locally whenever possible so that no bandwidth is taken up whenever possible
- Combiner functions can be used to combine intermediate key/value pairs prior to being sent over the network to increase speed of operation
- Deterministic crashes can be skipped if bugs prove too difficult to fix
- Users can add custom input data types by means of a *reader* interface
- Users can add counters to the Map or Reduce functions to count various actions
- Check points are kept track of in the event of a master computer fail so that the master computer can be restored
- The master computer pings the workers periodically to check for failures and reschedules tasks in the event of failures
- Backup tasks are run incase “stragglers” are slowing down the process due to a faulty disk or something of the sort

MAPREDUCE: ANALYSIS

- MapReduce is an excellent tool for data analysis both for large scale businesses and up and coming ones with little experience. MapReduce has a number of features that make the life of the user much easier. Not having to worry about the details of fault tolerance and parallelization is a great thing for novice programmers who need to do this sort of large scale data processing. Large businesses can also take advantage of these features because they are able to make changes to their software at a faster pace in an ever competitive marketplace that demands the best from its most reputable businesses.

COMPARISON PAPER: MAIN IDEA

- MapReduce vs Parallel Database Management Systems (DBMS)
- Parallel DBMSs performed better in nearly every aspect when compared to MapReduce, but not all
- There were a number of issues that arose during the setup of the DBMSs that were not present during the setup of MapReduce
- MapReduce may be simpler to set up, but DBMSs may be worth it in the long run
- Both MapReduce and DBMSs are advancing and are slowly taking on the positive aspects that the other possesses

COMPARISON PAPER: IMPLEMENTATION

- DBMSs performed better than MapReduce in all aspects except for load times
- DBMSs can operate directly on compressed data
- MapReduce does not have a structured schema for input data types, so while this does allow for freedom it puts the burden of implementing the datatype on the user
- MapReduce users must also implement their own indexing
- Implementation of custom code for MapReduce needs to be accounted for when work is being shared between multiple programmers to avoid errors or confusion
- DBMSs can reorganize input data at load time due to its structured nature which allows unnecessary data to be ignored and never read, increasing speed
- MapReduce has simply implemented fault tolerance that is more efficient than DBMSs as well, due to MapReduce being able to restore mid query rather than having to restart the entire query in the event of a failure
- Much less coding may be required for DBMSs due to the ability to use SQL which is designed for data processing
- Efforts to integrate DBMSs and MapReduce are being worked on to combine the strengths of both through projects like Greenplum and Asterdata, although they are not mainstream yet

COMPARISON PAPER: ANALYSIS

- It seems that DBMSs are the better option in most cases. The initial ease of setup that comes with MapReduce does not seem worth the added trouble of long processing times, more coding in order to get things working correctly, and more coding just to keep your software up-to-date with your current data. If you plan on processing data for any extended period of time, the upkeep of MapReduce and less efficiency of it is simply not worth it even with the easier first time setup and better fault tolerance.

COMPARISON OF COMPARISON PAPER AND MAPREDUCE

- MapReduce is excellent at hiding the details and under-the-hood functions that are required in large scale database management, but simply does not out perform DBMSs where it counts. Processing speed and long term upkeep are two very important aspects of large scale database management, and they are two that MapReduce lacks when compared to DBMSs. Even when MapReduce comes out on top, like in terms of fault tolerance and customizability, it still suffers as a result. Fault tolerance comes at the cost of processing speed due to needing to write data at intermittent intervals during the analysis process and customizability comes at the cost of ease of use in terms of programming.

STONEBRAKER TALK: MAIN IDEA

- Common DBMSs are out of date
- Tried to encompass every possible thing, based on row stores and relational model
- Column stores are one of many faster
- Arrays vs tables, arrays are the future
- Streaming won't have row stores, graph analytics won't have row stores…
- Major companies are already starting to phase out old models, and common row store relational DBMSs are simply inefficient compared to other options that are being researched

MAPREDUCE IN CONTEXT OF COMPARISON PAPER AND STONEBRAKER TALK

- MapReduce and the relational DBMSs talked about in these papers may just be stepping stones in a bright future for database research. Compare databases to another field, astronomy. People thought they had everything figured out when they decided the sun revolved around the earth. Obviously this was wrong, and humans realized this and moved on to greater ideas. It may have made sense at the time, but as new information is discovered, as new advancements are made, people adapt. These relational row based models have allowed humans to grow and advance as a species at an incredible rate, and their importance cannot be understated, but it is also important that we do not hold ourselves back when opportunity knocks on our door.