Paper Summary:
MapReduce: Simplified Data Processing on Large Clusters

Nikol Pettine

11/16/2013

Work Cited

Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters." OSDI 2004. (2004) Web. 16 Nov. 2013.

Main Idea of MapReduce

- MapReduce - program created to read and create large data sets
- “...a simple and powerful interface that enables automatic parallelization and distribution of large-scale computations…” (pg. 1)
- Usually used on large cluster of PCs connected with switched Ethernet

MapReduce Implementation Steps

1. Splits files into many pieces of 16-64MB each, then starts many copies of the program on many machines.
2. 1 copy is the master, who assigns either a map/reduce task to the workers
3. Map task reads file and parses key pairs out and buffers them in memory
4. Buffer pairs written to a local disk, location sent to the master
5. Reduce task reads buffered data and sorts by keys, so all values who have the same key are grouped together
6. Passes key with corresponding set tp user's Reduce function, then it is appended to the final output file
7. Master, once all tasks are finished, ends MapReduce call and returns to user's code

Analysis

- Creates a simple way to do large computations without dealing with a lot of complex code
- Implementation meant to be fault tolerant of any type of failure, skips bad records that could cause errors, and made to be simple and quick, on a large-scale of computers

Advantages of MapReduce

- Simple, small, understandable code
- Handles failures and errors automatically
- Input data read locally, so uses no network bandwidth
- Small amount of workers for a large amount of tasks
- Backup, redundant, executions to reduce total time for a large MapReduce operation

Disadvantages of MapReduce

- If master fails, though very unlikely, will abort the MapReduce
- No support for “atomic two-phase commits of multiple output files produced by a single task.” (pg. 7)
  - has not been an issue, yet
- Map functions are hard to code

Real-World Use Cases

- Count of frequency to certain URLs
- Inverted index
- Distributed sort
- Google web searches
- Data mining