Worksheet #1

Parallel Databases x Map-Reduce April 24th, 2019

Based on the following papers:

1. [Pavlo et al., SIGMOD 2009] A Comparison of Approaches to Large-Scale Data Analysis
2. [Dean and Ghemawat, CACM 2010] MapReduce: A Flexible Data Processing Tool
3. [Stonebraker et al., 2010] mapReduce and Parallel DBMSs: friends or foes?

Questions

1) Does it make sense comparing map-reduce approaches with parallel and distributed database systems? Explain your answer.

Yes it does, both techniques try to entail the harnessing of parallel work in order to solve computing problems and provide a means to process large volumes of data. There are contexts and problems were both approaches can be used, each one bringing its advantages and disavantages to the problem at hand - Let's take as example the problem of dealing with the enourmous amount of data that Facebook collects [3].

2) What are the advantages of map-reduce over parallel and distributed databases?

There are several advantages of map-reduce over parallel and distributed databases, namely:

- **Inferior data loading time**, since there isn't any kind of preprocessing of the data, whilst there is on parallel and distributed databases;
- **More fault-tolerant** then its counterpart, as seen in [1];
- **Open-source implementations**, while the major set of parallel and distributed databases are very expensive;
- **No schema is enforced**, as the data does not need to follow any kind of schema which can be good in cases of semi-structured data.
- **Allows for more complex data interrogation** while in parallel and distributed databases the SQL language can be a restriction in the type of queries that can be made.
- **Excel at complex analytics and ETL tasks**.

3) What are the advantages of parallel and distributed databases over map-reduce?

There are several advantages of parallel and distributed databases over map-reduce, namely:

- **Use of indexes** greatly improving query time when compared to the map-reduce approach;
- **Works with compressed data**, making the nodes access to data faster;
- **Automatic optimizations in task distribution** - the system would distrbute tasks in such way that the flux of data between nodes is minimized;
- **Schema is enforced** which pushes programmers to a higher, more-productive level of abstraction.
- **Faster data interrogation**, complex queries in the map-reduce approach are simplified by the usage of the SQL language.

4) What kind of operations are allowed in parallel and distributed databases that are not available “out-of-the-box” in map-reduce?

Represented as subset of the advantages previously presented, the operations that are allowed in parallel and distributed databases that are not available “out-of-the-box” in map-reduce are:

- **Use of indexes**;
- **Automatic optimizations ins task distribution**;
- **Works with compressed data**;
- **Schema**.

5) What kind of operations are allowed in map-reduce that are not available “out-of-the-box” in parallel and distributed databases?

Represented as subset of the advantages previously presented, the operations that are allowed in map-reduce that are not available “out-of-the-box” in parallel and distributed databases are:

- **Fault-tolerance**;
- **No Schema**.

6) Are there alternatives to Google’s mapreduce? How do they perform?

Some alternatives to Google's mapreduce are:

- **Apache Hadoop**;
- **Apache CouchDB**;
- **Disco**;
- **Infinispan**;
- **Riak**.

In general they perform Ok. Apache Hadoop is the most popular at the time of writing.

7) Search the web for parallel and distributed databases. In which situations would be interesting to use such solutions instead of using google’s mapreduce solution?

Vertica - a commercial column-store relational database; DBMS-X - a row-based database from a large comercial vendor. For the following types of problems, it's suggested that parallel DBMS should be chosen above MapReduce:

- The data is structured and will continue to be so for the foreseeable future.
- The data set is large and is expected a large amount of complex querying.

Authors

- Afonso Pinto
- Edgar Carneiro