Big Data Analyses

Megan Poyntz

May 8th, 2015

A Comparison of Approaches to Large-Scale Data Analysis

D. Abadi, D. DeWitt, S. Madden, E. Paulson, A. Pavlo, A. Rasin and M. Stonebraker. "A Comparison of Approaches to large-Scale Data Analysis." 2009.

Hive – A Petabyte Scale Data Warehouse Using Hadoop

S. Antony, P. Chakka, N. Jain, H. Liu, R. Murthy, J. Sarma, Z. Shao, A. Thusoo and N. Zhang. "Hive – A Petabyte Scale Data Warehouse Using Hadoop." 2010.

Hive: Main Ideas

* Open-source data warehousing solution built on top of Hadoop
* Further developed than basic MapReduce models
* Supports declarative-language queries (like SQL queries) that are executed in similar ways to those in the MapReduce model → language is called HiveQL
* Supports customizable MapReduce data execution queries
* Parallel DMBS-like structure for some aspects (e.g. Metastore, arrays & maps)
* Flexible infrastructure that helps with large-scale data analytics, business intelligence and reporting tasks

Hive: Implementation

* Stores data in a DBMS-like structure: tables of rows and columns and partitions
* Supports DBMS-like data types, as well: easy types (integers, floats, etc.) and sophisticated types (maps, lists, etc.)
* Tables are serialized and deserialized using default serializers and deserializers already present in the program
* If incorporating data prepared by another program, Hive has flexibility to implement it without having to transform the data, saving a substantial amount of time when loading large sets of data
* Queries can include clause subqueries, various join types, cartesian products, group bys, unions, creates, selects, etc.
* Data stored like regular tables, however table metadata stored into hdfs directories → consist of tables, partitions, and buckets

Hive: Analysis

* Program is imperfect and has a long way to go (e.g. can only implement *some* SQL/ declarative-language-like queries, but not all)
* Cannot fully integrate with commercial Business Intelligence programs that support relational-type data warehouses
* Has proved helpful because of the increasingly popular principle of having flexibility when building a database (e.g. the freedom to customize that the MapReduce framework allows)
* Useful for users that are already familiar with SQL → they can write and execute queries with their prior knowledge, as well as easily understand queries written by other users. Helpful if using across a business where data is shared constantly

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

* With an increase in cluster computing to solve computing problems, users want a simple model that both loads data quickly and gives users the ability to express somewhat sophisticated programs

* MapReduce is becoming increasingly popular among users because it provides the above for cluster computing

* There are pros and cons for both MapReduce and parallel DMBSs: they depend on the system itself and what the database user is looking for

* A user's choice of database model and system should depend on what they prioritize in their database usage (e.g. data loading, analyses execution, etc.)

A Comparison of Approaches to Large-Scale Data Analysis: Implementation

MapReduce

* 2 functions: Map & Reduce
* Map → reads set of records from an input file
* Reduce → data transferred from Map nodes' local disk
* Data split into N pieces across many computers
* Map function writes output to local disk, Reduce function reads data and writes to output file

Parallel DMBSs

* Data required to fit into structure of rows and columns
* Tables are partitioned over nodes in a cluster
* System uses an optimizer that translates commands into query plan
* Execution is divided amongst multiple nodes

A Comparison of Approaches to Large-Scale Data Analysis: Analysis

* Parallel DMBSs require well-defined schema, whereas MapReduce permits arbitrary data format
* MapReduce is an ideal approach to large amounts of data that do not require strict structure nor require much time to load the data
* Hadoop (and therefore other MapReduce modeled systems) proved easier to set up and use
* Parallel DBMSs, being more developed, perform significantly better in executing data analyses than MapReduce Hadoop

Ideas & Implementation Comparison

* Both Hive and Hadoop support customization to certain extents (e.g. schema customization, languages)
* Structure from parallel DMBSs has proven helpful. It is the reason parallel DBMSs have been so successful for tens of years, and the reason Hive receives so much support
* Both Hive and MapReduce models have flexibility in many areas for users. Being that flexibility is ideal, they are both becoming increasingly popular and in-demand systems among many big data-computing users
* The structure found in DBMSs and (to a point) in Hive have proved helpful for businesses, especially with consideration of business intelligence systems for traditional relational data warehouses
* Executing queries in SQL and HiveQL are extremely similar → provides consistency among various data models and programs that is helpful for computing users across different companies and industries
* Hive has the flexibility to implement different data types, again providing freedom and customization to its users, just like MapReduce-modeled programs

Advantages & Disadvantages

Advantages

* **MapReduce** → arbitrary structure allows user's schema preference (choice of some sort of structure, or no structure at all)
* **DMBSs** → structured and pre-determined format that's consistent and easy to implement for everyone
* **Hive** → customizable (to an extent) while having relational database-type attributes (e.g. SQL/declarative-language queries)
* **HiveQL** → easy for users that are familiar with SQL to understand and write in

Disadvantages

* **MapReduce** → arbitrary structure means its time consuming to create and implement schema if you need one (especially if you are sharing it)
* **DMBSs** → structured model cannot allow users to customize schema if they need to change it
* **Hive** → under-developed (cannot execute all SQL/declarative-language-like queries, cannot integrate with all relational-model Business Intelligence systems, etc.)