Alex Hochberg
May 9, 2014

Bigtable: A Distributed Storage System for Structured Data
&
A Comparison of Approaches to Large-Scale Data Analysis

A. Pavlo, E. Paulson, A. Rasin, D. J. Abadi, D. J. DeWitt, S. Madden, and M. Stonebraker, "A comparison of approaches to large-scale data analysis," in *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*, New York, NY, USA, 2009, pp. 165-178.

F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, R. E. Gruber . "Bigtable:A Dstributed Storage System for Structured Data" in *ACM Transactions on Computer Systems (TOCS),* ACM New York, NY, USA , 2008, Volume 26 Issue 2.

Main Idea of *Bigtable: A Distributed Storage System for Structured Data*

- Bigtable is a compressed distributed storage system for managing structured data
- Designed for high performance and to scale a very large size (petabytes of data)
- Data model is a sparse, distributed, persistent multidimensional sorted map
- Gives clients dynamic control over data layout and format, as well as reason about the locality of data in storage
- Used exclusively within Google by 60 products/projects including Google Finance, Google Earth and Personalized Search
- Built upon Google File System, Chubby Lock Service and SSTable

Bigtable Implementation

- Map is indexed by a row key, column key and timestamp (which allows three-dimensional mapping)
- Row keys = atomic arbitrary strings maintained in lexicographic order. Row range is called *tablet*
- Columns grouped in sets called column families. Column key's syntax = *familiy:qualifier*
- Timestamps are recorded in real time, multiple versions of same data, stored in decreasing order so most recent is first
- Implementation has three major components:
- (1)A library that is linked into every client
- (2)One master server; responsible for assigning tables to tablet servers
- (3)Many tablet servers; can be dynamically added or removed from cluster

Analysis of Bigtable and Implementation

- Effectively handles large scale data by using the BMDiff and Zippy algorithm to compress tables when size threatens to grow beyond its limit
- Maintains reliable and consistent tablet servers by using the Chubby Lock System
- Performance of Bigtable read/writes per second increase by a factor of 300 as the number of tablet servers increase by a factor of 500 (almost linear)
- Offers customizable and relevant client control of data through the decision of locality properties of data
- Timestamp index provides continuous records of different versions of data with time, an accommodation not offered in all database designs
- Uses in real world show that new users find the Bigtable interface hard to adapt to

Comparisons within A Comparison of Approaches to Large-Scale Data Analysis

- Recent attraction to the MapReduce computing model is its simplicity; consists of two functions Map (which does filtering and/or transformations) and Reduce(which summarizes and record)
- When compared to SQL DBMSs, that have been in place for the last 20 years, MapReduce proved to be significantly slower and required more code to implement each task, but took less time to tune and load data
- Parallel DBMSs support the relational model is criticized for requiring data to adhere to its rigid schema, although this model succeeds in allowing faster and easier to write queries
- MapReduce does prove to be quite flexible but MR programmers are still required to work to build whatever structure exists for input files
- DBMS's programming model of "stating what you want," rather the MR Codasyl's "presenting an algorithm for data access" proved to make programs easier to write, modify and understand

Advantages and Disadvantages of Bigtable in Context of A Comparison of Approaches to Large-Scale Data Analysis

- If high availability of more recent versions of data is desirable and applicable in your database system, Bigtable's timestamp index proves to be useful advantage over DBMS (especially within the Personalized Search program)
- DBMS's relational model is easy to understand and implement for users, as compared to Bigtable's difficult to use interface
- The DBMS model ensures the maintenance of integrity, control of redundancies, and avoidance of inconsistency
- Both Bigtable and DBMS (through its process of compression) are able to handle large scale data and maintain high performance
- The newness of Bigtable makes it vulnerable to many types of failure, especially when adding new features, while the relational model proves once and again to be a simple and effective design