Bigtable: A Distributed Storage System for Structured Data

Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach

Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber

Brian Canoni

5/8/14

The main idea of Big Table

- Big table was developed to be a distributed system to store and manage data on a very large scale.
- It is meant to handle systems such as Google Earth and website Indexing. It supports both a wide range of flexibility and high scalability yet it also maintains high speed performance.
- Data is distributed and even with petabytes of data Big Table reliably stores this information is widely available and is used in more than 60 different google products.

The Implementation of Big Table

- The Data is stored and organized in terms of a multi-dimensional map and are indexed by things such as timestamp and row/column key.
- A custom API(application programming interface) was developed to utilize this with similar functions to a relational database such as adding, changing, restricting information and metadata.

Implementation Cont.

- Big table uses the distributed Google File System (GFS) to physically store data files and logs.
- Big table clusters usually use a shared pool of machines which run Big table processes. It utilizes a cluster management system to schedule jobs because these machines are typically already doing jobs for other systems and must be shared.

Implementation Cont.

- Big table to maintain data integrity uses a system called Chubby which is a distributed lock service just like the ones used in typical relational databases but designed for use in Big table.
- This Chubby system ensures that at any one time there is at a maximum only one master.
- Additionally Big Table is entirely dependent on chubby and if the Chubby system is not working Big Table will not work.

Implementation Cont.

- Data is stored in Tables like a relational database but each table is made up of a set of “tablets” (tab – lets like piglets) .
- Each of these tablets contains all of the data that is linked to a certain row range .
- Upon creation each table will start with a tablet and as rows are added this will split into new rows.
- This system has three levels of data. The first being the root tablet, the second is the metadata tablet, and the last are user tablets.

Implementation Cont.

- Other performance enhancing features such as Compression, Caching, and Bloom filters are implemented in Big Table. Additionally with log file creation separate log files are created per tablet and mutations are added to the larger commit log on each server.

Analysis

- I think that Big Table is a good idea in theory. It seems to mimic a lot of the concepts of a traditional relational database but allowing for a lot more flexibility. This really shows in its “tablet” idea where tables are made up of several metadata tablets which store the location of other data tablets.
- Presumably that entire processes would speed up finding data as the data is stored like a tree and instead of searching through all the leaves the branches would already know the exact location of each leaf.

Analysis Cont.

- Also I thought that having tables broken up like that would make locking much easier especially because their locking system Chubby is specifically designed for Big Table.
- Utilizing the physical storage system GFS big table really only has to do with the logical formation of tables. This complicates the process of data storage in my opinion as two different systems must be used to actually store and access the data.

Analysis Cont.

- Although this system is complicated in design and really doesn't change too much conceptually compared to a traditional relational database, I can see how it would be helpful in systems that require petabytes of information to be stored and accessed reliably and quickly.

Comparison to Comparison Paper

- Compared to the several other approaches to big data Big Table seems to be organized very similarly.
- Big Table still utilizes rows and columns technically even if these rows and columns are broken up and spread out, the functionality still remains the same.
- It is indexed through metadata tablets which is much more complex than normal but it accelerates access to data like other modern DBMS.
- The article talked about SQL's limited "expressive prowess" yet it still being a powerful tool. Big table stated that it has its own custom API to operate with which from its description supports all of features of typical SQL but adds to that with more expressive uses.

Comparison to Comparison Paper

- The structure of Big Table allows for several processes to be run in parallel. It uses clusters which each pass on changes to the data higher up so that data is consistent. Compared to other systems Big table's cluster manager seemed to be a similar idea.
- Performance wide compared to other large scale data systems. The google paper states that at around 250 nodes the performance doesn't increase quite as much per node beyond that point. This was true for scans, reads, and writes. This also proved true in other big data systems where performance slowed down as more nodes were added past a certain point.

Advantages

- Big Table provides a lot of flexibility by having its own API as well as its own locking system Chubby. These provide Big Table with the power of tools like sql but improve upon it by catering to the data needs of Google.
- Also breaking the tables down into “tablets” makes the process of locking data for reading and writing much faster especially when combined with Chubby.

Disadvantages

- Unlike other modern day systems Big Table actually only has to do with the formation of tables. The actual physical writing part is done by Googles GFS. This makes Big Table dependent on GFS.
- Because Big Table uses many clusters all doing processes in parallel it reportedly begins to slow down when a large number of processes are changing things at once.
- Big table also stated its dependency on its locking system Chubby which is a separate system which could go down. Due to the possibility of Chubby failing and breaking the whole Big Table process has another possible point of failure.