Bigtable: A Distributed Storage System for Structured Data

Summary:

Bigtable is a distributed storage system for managing structured data that is designed to scale to a very large size: petabytes of data. Many projects like web indexing, Google earth and Google finance store data in Bigtable.

Introduction:

Google took almost three years to design, implement and deploy a storage system for managing structured data called Bigtable. It is designed to reliably scale to petabytes of data and on thousands of machines. It has achieved several goals: wide applicability, scalability, high performance and high availability. It is like normal database in many ways but also differs in interface from normal relational database. It provides clients with more control over the data model and format.

Data Model

A Bigtable is a sparse, distributed, persistent multidimensional sorted map. The map is indexed by a row key, column key, and a timestamp; each value in the map is an uninterpreted array of bytes.

(row: string, column: string, time:int64) --> string

Row: Bigtable maintains data in lexicographic order by row key. The row range for a table is dynamically partitioned. Each row range is called a tablet, which is the unit of distribution and load balancing.

Timestamp: Each cell in a Bigtable can contain multiple versions of the same data; these versions are indexed by timestamp. Bigtable timestamps are 64-bit integers. They can be assigned by Bigtable, in which case they represent real time in microseconds, or be explicitly assigned by client applications.

Columns: Column keys are grouped into sets called column families, which form the basic unit of access control. All data stored in a column family is usually of the same type (we compress data in the same column family together). A column key is named using the syntax: family: qualifier. Column family names must be printable, but qualifiers may be arbitrary strings.

API

Bigtable API provides clients with ability to creating and deleting tables and column families. Client can write, delete, lookup or iterate over the values of the data in the table. Bigtable can also be used with MapReduce.

Building Blocks

Bigtable is built on several other pieces of Google infrastructure.

Bigtable uses the distributed Google File System (GFS) to store log and data files. Bigtable depends on a cluster management system for scheduling jobs, managing resources on shared machines, dealing with machine failures, and monitoring machine status The Google SSTable file format is used internally to store Bigtable data. An SSTable provides a persistent, ordered immutable map from keys to values, where both keys and values are arbitrary byte strings. Bigtable relies on a highly-available and persistent distributed lock service called Chubby. A Chubby service consists of five active replicas, one of which is selected to be the master and actively serve requests. The service is live when a majority of the replicas are running and can communicate with each other.

Implementation

The Bigtable implementation has three major components: a library that is linked into every client, one master server, and many tablet servers. Tablet servers can be dynamically added (or removed) from a cluster to accommodate changes in workloads.

The master is responsible for assigning tablets to tablet servers, detecting the addition and expiration of tablet servers, balancing tablet-server load, and garbage collection of files in GFS. In addition, it handles schema changes such as table and column family creations. Each tablet server manages a set of tablets (typically we have somewhere between ten to a thousand tablets per tablet server).

The tablet server handles read and write requests to the tablets that it has loaded, and also splits tablets that have grown too large. As with many single-master distributed storage systems, client data does not move through the master: clients communicate directly with tablet servers for reads and writes. Because Bigtable clients do not rely on the master for tablet location information, most clients never communicate with the master. As a result, the master is lightly loaded in practice.

A Bigtable cluster stores a number of tables. Each table consists of a set of tablets, and each tablet contains all data associated with a row range. Initially, each table consists of just one tablet. As a table grows, it is automatically split into multiple tablets, each approximately 100-200 MB in size by default.

Conclusion

As of August 2006, more than sixty projects were using Bigtable. Users like the performance and high availability provided by the Bigtable implementation, and that they can scale the capacity of their clusters by simply adding more machines to the system as their resource demands change over time. Google was also in the process of implementing several additional Bigtable features, such as support for secondary indices and infrastructure for building cross-data-center replicated Bigtables with multiple master replicas. They have also begun deploying Bigtable as a service to product groups, so that individual groups do not need to maintain their own clusters.