Paper Evaluation

Bigtable: A Distributed Storage System for Structured Data

Michael Lanthier

Google's Bigtable is a storage system designed to scale to large amounts of data and provide high performance by exploiting data locality. Bigtable is effectively a multi dimensional sorted map that indexes on a row key, column key, and a timestamp with the data being stored as a array of bytes. This data is split into tablets, a range of rows that are kept together to best distribute the data. Bigtable uses a distributed lock manager called Chubby and a master server to distribute these tablets to individual servers called tablet servers which receive reads and updates from clients. The tablet servers ultimately write to Google File System to store the data. By keeping similar data close to one another finding the location of where to read and write is fairly cheap and allows for effecient querying of large amounts of data. Bigtable is able to effectively store up to petabytes of data as proven in active systems using it as well as scale well with reads and writes in test systems.

Questions/Comments

1. How do we know when to spin up a new master? I assume when the master decides to kill itself it can trigger that but what if the master goes down? Do the tablet servers need to identify that master can no longer be reached and then spin up a new one. The system can still do work while the master is still down as all it does is ensures that tablets servers are assigned but if it never comes back and tablet servers start to die their tablets will never be re-assigned.
2. Compaction of the memtabe looks exactly like it does in LSM trees. It even compacts SSTables just like LSM trees as well.
3. When the clients cache is stale it may take up to six round trip times. That is a lot of time especially if you are making requests from a great distance or if there was network delay. The paper goes on to say that this can be reduced by prefetching tablet locations. How much does this actually help? Will the first time they try and get new tablet information on a stale request still take six RTT and then after that it will be less or none? In their testing they say all the machines are in a single host facility so if the client is making requests say from the other side of the globe, this could be a quite expensive operation.
4. Their testing appears to be have done in a single hosting facility. Is there a way to

distribute where the tablet servers are outside a single facility or would this just be too costly? It seems like maintaining the tablet servers and tablets with the master and Chubby is a fairly chatty process so increasing network time may not be very practical. In the case of large scale failure at the hosting facility though the system becomes inoperable. Is there an effective way of doing Bigtable such that it can withstand large scale failures similar to that of Amazon Aurora?