Scott Viet Phong Nguyen, 350313

Anh Duc Nguyen, 379744

Task 1 – Misc. (10+10+10 = 30%)

a) Explain with your own words how distributed storage engines (like Dynamo) is handling temporary failures and give an example.

- Technique: "Sloppy Quorum" and hinted handoff
- these techniques provide high availability and durability guarantee when some of the replicas are not available
- all read and write operations are performed on the frist N healthy nodes from the preference list, which may not always be the first N nodes encountered while walking the consistens hashing ring
- with an example DynamoDB configuration of N=3 and a temporary failure occurs or a write operation is happening on node A, a replica that would normally have lived in A will now be sent to node D. The replica will have a hint in its metadata with a suggestion to the node which was intended - A. The hints are kept in a separate local databse that is scanned periodically. If A is available again, D will attempt to deliver the replica to A. After the transfer, D may delete the object from its local store

b) Give a reasons why one should use distributed caching and explain it in a use case.

- For cases in which the number of objects that can be or should be cached is unpredictable and large, and consistency of reads is a must-have
- For example - online catalogues: end users are likely to access the same data a lot of times during the course of the day. Fast access to item prices and other information are crucial for the satisfaction of the user experience for this web application. Ensuring good performance through making that data available in a coherence and distributed cache is the solution.

c) Name three advantages of distributed caches and explain them in one sentence.

- distributed cache can be bigger in cache size, it combines resources from several servers
- distributed cache can reduce the cost of lookups across all the servers in the cluster, because distributed cache provides an architecture where all the hosts are sharing the load of answering reads requests
- distributed cache limits the interruption to typical operations during optimization and maintenance activities, so performance-enhancing tools don’t break continuous operations and performance tanking is avoided

Task 2 – GFS (20+10+10 = 40%)

a) Which operations are supported by GFS? For which types of queries is GFS optimized?

GFS's Interface supports the following operations: create, delete, open, close, read and write. There are also two more operations snapshot, which creates a copy of a file, and record append, which allows multiple clients to append data to the same file concurrently while guaranteeing the atomicity of each individual client's append.

The client sends chunk index query requests to the master. It does it by translating file name and byte offset specified by the application into a chunk index within the file. The master sends the corresponding chunk handle and locations of the replicas and the client caches this information and contacts one of the replicas for data.

b) State two main differences between Dynamo and GFS with regard to operation and data types?

In Dynamo smaller data elements of file pointers are saved from 1 byte up to 400KB, however in GFS there are several chunkservers which contain files divided into fixed-sized chunks, usually 64MB. Dynamo supports GET/PUT operations using a user-defined primary key to read or write the indexed structured data. In GFS however the Clients interact with master for metadata operations, but with chunkservers for data operations.

c) Explain why GFS is particularly well suited as storage system for MapReduce job input and output data (why not Dynamo or a relational database?).

GFS is particularly well suited as storage system for MapReduce job input and output data because the MapReduce model is an implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster. Since Dynamo or Relational Databases can store small data they are not suitable. On the other hand GFS stores large data on different servers and therefore is well suited.

Task 3 –BigTable (20+10=30%)

a) Please describe in your own words how the BigTable storage engine works by explaining the steps of a write request, and a read request, respectively (local persistence).

If updates are recently committed, these are stored in a sorted buffer called "memtable". Older updates are stored in a sequences of SSTables. A tablet contains metadata, which contains a list of SSTables and a set of pointers, which points into any commit logs that may contains data for the tablet. The server reads the indices of the SSTables into memory and reconstructs the memtable by applying all of the updates that have commited since the pointers.

When a write request arrives at the tablet server, the server checks whether it is well-formed and that the sender is authorized to perfom a mutation, which are done by reading the list of permitted writers from a Chubby File. Valid mutations are written to the commit log and after the write has been commited, its content is inserted into the memtable.

With a read request for the tablet server, the server also checks for the well-forming criteria and authorization first. Then the valid read operation merges the sequence of SSTables and the memtable. The merged view is formed very efficiently, because the SStables and the memtables are lexicographically sorted data structures.

b) True or False: Bigtable provides additional security, geo-distribution and support for big files. If False, shortly explain.

False, Bigtable only enhaces GFS with a structured data model and more sophisticated query functionality. Moreover it additionally supports small files, while maintains system properties like high availability, high troughput, low latency and sub-linear scalability.