Casper Kristiansson
2023-09-06

Data Intensive Computing – Review Questions 1

1. **Explain how a file region can be left in an undefined state in GFS?**

Namespace mutations or mutations in GFS are actions such as creating a file or modifying a file. These types of actions are atomic which means that they either get completed successfully without any issues or they don't go into effect. This means that when these types of mutations happen the state of a particular region gets labeled with either **consistent, defined,** or **undefined.** Consistent state is when the data exists the same on all replicas which means that the client can read from any of them. A defined state is when the data in a region is not only consistent on all replicas but also contains the entirety of what the mutation wrote to a region. **An undefined** state is when a region may not reflect a specific mutation for example if two mutations happen at the same time the resulting data could be a mix of the mutations. An undefined state will also happen if the mutation fails, and the region becomes inconsistent because the different replicas will have different data.

2. **Briefly explain the read operation in GFS? What's the role of the primary node?**

When a client wants to read a file or a specific part of a file the client will calculate which chunks it wants. The client will then ask the GFS master for the information regarding the chunk. The master will then identify which chunk servers that hold the information. The master will then respond with the location of different replicas of the requested chunk. When the client receives this data, it will in most cases cache it to avoid sending the same request to the master. After this, the client will read data directly from one of the chunk servers that was provided. Because the client has a list of multiple chunks servers if one fails the client can always choose another one on the list to handle failures. After each request, the chunk server will send the requested data to the client.

The role of the primary node has a couple of main objectives. The first is the lease holder which the primary will hold the leases for particular chunks. The master will grant these leases to particular chunks so different chunk servers become lease holder for the chunk. The second is mutation/operation serialization where the goal is to assign the order the different operations should be executed. One of the main reasons to do this is to make sure that all replicas are consistent with each other. The third reason is mutation propagation which is the process of making sure that all changes made to the lease chunk are replicated to all secondary replicas. Another role of the primary node is error handling where the primary will communicate the status of mutations back to the client. The goal of this is that if an error occurs the client can handle the mutation error.

3. **Using one example show that in the CAP theorem, if we have Consistency and Partition Tolerance, we cannot provide Availability at the same time.**

Casper Kristiansson
2023-09-06

CAP theorem states by Eric Brewer that a distributed system can provide at most two out of three guarantees simultaneously of consistency, availability, and partition tolerance. Consistency ensures that the data is always up to date meaning that read operations will always retrieve the most recent write. Partition tolerance is the process that if a communication loss occurs between two parties the whole system will continue to function. Meaning that if one link fails and certain parts of the system cannot communicate with each other the system can continue to work.

A basic example to determine that if a system has consistency and partition tolerance it cannot have availability at the same time is for example with a bank system. Let's say a bank has multiple data centers where it stores all data regarding its bank like transactions balances etc. If a customer deposits X amount of money at one bank (one data center) the data center will send the update to all the other data centers. If one of the updates fails due to network issues the data will not be consistent. In this process, a partition will be created. To battle this the system will not acknowledge or allow any new deposits or transactions until the partition has been resolved to uphold the consistency. But because of this, the availability of the system will be affected.

4. **Explain how the consistent hashing works.**

Consistent hashing is the process commonly used in distributed systems to distribute data to multiple locations in a way to minimizes the process of remapping or moving data when nodes are added or removed.

Consistent hashing uses something called a hash ring (visualization tool). On the ring, there exist different nodes spread out. When a data needs to be assigned a hash value gets associated with it and placed on the ring. The has value originates from a has function and its purpose is a mapping of the data and the positions of the nodes on the ring. When placed it will move clockwise until it encounters a node where the data will be placed/located. By doing this when a new node needs to be added the data items in the specific segments need to be moved/associated with the new node. Rather than needing to remap or move a lot of data.

The process of removing nodes works by simply removing it from the ring and all data that was associated with it will continue to go clockwise until it finds a new node. Virtual nodes are something that is also a key aspect of consistent hashing. The goal of virtual nodes is to handle load distribution. Instead of assigning a node one position on a ring virtual node helps to assign multiple positions.

5. **Explain the finding process of tablets in BigTable.**

BigTable is a concept in distributed storage systems for managing data that was introduced by Google. The basics of BigTable is the process of dividing data into tablets and finding a way to find the correct tablets efficiently.

A table in BigTable is divided into multiple smaller tablets. Each tablet represents a range of rows. To keep track of what rows are stored in each smaller table meta tables are

introduced. The goal of them is to keep track of tablet and their locations. When trying to find tablets the system is structured in a three-level structure. This means that there is first a root tablet which contains the location of the second level of meta tablets. The second level of meta-tablets in turn contains the location of the actual tablets where the data is located (row). This means that to find any specific row the system will need to query all three tablets to get the actual data.

A big thing to note is that caching in this kind of system is important to improve the multi-level lookups. By doing this a client can bypass the multi-level lookup and can read directly to the relevant tablet.