CSE535: Distributed Systems (Fall 2021)

Scott D. Stoller, Stony Brook University

Problem Set 2 (version 2021-11-26), Due 8 December 2021

INSTRUCTIONS: Do this assignment with your project teammates. **Justify your answer for every problem**, except where directed otherwise. All problems have the same weight. Exactly one member of each team should submit the assignment on Blackboard by 11:59pm on the due date. The pdf file should be formatted for 8.5"×11" paper and should include the team name and names of team members at the top of page 1; also, the filename should include the team name. I recommend that each team member work individually on all problems, then the team can meet to pool everyone's thoughts and organize the final write-up. This will help everyone be better prepared for the exam.

Problem 1: GFS

Section 4.5 of the Google Filesystem (GFS) paper implies that chunk version numbers are incremented only when a lease is granted by the master, not when each mutation is serialized by the primary. Consequently, it seems that a secondary that successfully increments its version number for a chunk, and then misses a mutation because of communication failures or a crash failure, could still have the correct version number for the chunk. How is this issue dealt with in GFS? For example, what would prevent a client from reading a chunk from such a secondary?

For simplicity, consider only write operations. Ignore append. The situation for append is complicated by GFS's weak semantics for append (recall that GFS re-tries a failed append at a different offset).

Problem 2: Bigtable

Write high-level pseudo-code for the tablet location lookup algorithm in the Bigtable client library. The inputs to the algorithm are the table and row key. The output is the identity of a server with a replica of the tablet containing the specified row of the specified table. Remember that some relevant information might already be cached at the client. Your pseudocode should reflect failure handling when a queried server does not reply. Consider only failure of tablet servers; do not consider failure of the network, the master, or Chubby.

Problem 3: Megastore

This question is about Megastore. (A) As part of processing a snapshot read request, does a replica need to ask its coordinator whether it is up-to-date for the entity group being read? If the paper directly addresses this question, your answer should quote the relevant passage. Otherwise, you should propose an answer and explain why it is the most reasonable approach. (B) Describe different kinds of scenarios in which a snapshot read and a current read would return different results. Describe as many different kinds of such scenarios as you can think of. Hint: there is at least one kind and at most two kinds. Note that a scenario should be described as a *sequence of events*.

Problem 4: Chord

Write pseudo-code for n.stabilize(), n.find_successor(id) and n.closest_preceding_node(id) for the version of Chord that uses successor lists of length r. Include explanatory comments in your pseudo-code. Discuss any interesting design decisions, e.g., if there is a trade-off between the cost of stabilization and the speed with which knowledge of changes propagates through the system.

Problem 5: TAO

Consider a web server W assigned a primary follower tier T1 and a secondary follower tier T2 in a slave region R.

(a) What is the latency of a write by W to an object id_1 or an association $(id_1, type, id_2)$ without an inverse, in the absence of failures?

(b) Same as (a), except while the follower F in tier T1 responsible for id_1 is down.

(c) Same as (a), except while the leader L in region R responsible for id_1 is down.

(d) Same as (a), except for a write to an association with an inverse.

In each case, express the latency in terms of intra-region message latency L_0 and inter-region message latency L_1. Justify your answer by describing the relevant message sequence. Assume TAO servers and database (MySQL) servers are on different hosts. Ignore local processing time. Consider only the latency within TAO, i.e., start the timer when a follower receives the request from a web server, and stop the timer when a follower sends the response to the web server. For the cases involving failures, assume the failure has already been detected, so you do not need to consider the timeout used in failure detection.