The Andrew File System

Annalise Tarhan

March 26, 2021

1 Run a few simple cases on afs.py to make sure you can predict what values will be read by clients.

Seed 1
c0 write:0 value? → 1 value=0
c1 read:0 → value? value= 0
c0 read:1 →value? value=1
file:a contains: 2

Seed 2
c1 write:0 value? → 1 value=0
c0 write:0 value? → 2 value=1
c1 read:1 → value? value=2
c0 read:1 → value? value=2
file:a contains: 2

Seed 3, two files
c0 write:0 value? → 2 value=0
c1 read:0 → value? value=1
c0 write:1 value? → 3 value=2
c1 write:1 value? → 4 value=2
file:a contains: 3
file:b contains: 1

2 Now do the same thing and see if you can predict each callback that the AFS server initiates.

Callbacks are sent to a client when a file it has open is modified and closed by another client.

Seed 4
After c1 writes to and closes file a: callback: c:c0 file:a

Seed 5, 2 files
After c0 writes to and closes file b: callback: c:c1 file:b
After c0 writes to and closes file b: callback: c:c1 file:b (again)

3 Similar to above, run with some different random seeds and see if you can predict the exact cache state at each step.

Seed 6
c0: [a: 0 (v=1,d=0,r=1)]
c1: [a: 0 (v=1,d=0,r=1)]
c0: [a: 1 (v=1,d=1,r=1)]
c1: [a: 2 (v=1,d=1,r=1)]
c0: [a: 1 (v=0,d=1,r=1)]
c1: [a: 2 (v=1,d=0,r=0)]
c1: [a: 2 (v=0,d=0,r=0)]
c0: [a: 1 (v=1,d=0,r=0)]
c1: [a: 1 (v=1,d=0,r=1)]
c1: [a: 3 (v=1,d=1,r=1)]
c0: [a: 1 (v=1,d=0,r=1)]
c0: [a: 1 (v=0,d=0,r=1)]
c1: [a: 3 (v=1,d=0,r=0)]
c0: [a: 1 (v=,d=0,r=1)]

4 Now run the simulation with the following workload. What are the different possible values observed by client 1 when it reads the file when running with the random scheduler? Of all the possible schedule interleavings, how many of them lead to client 1 reading the value 1, and how many reading the value 0?

Depending on whether client 1 reads the file before or after client 0 writes to it, the observed value will be 0 or 1. Because the read and write operations occupy the same position in each client's operation sequence, the chance of one occurring before the other is 50%.

5 Now run the simulation with the same workload as before but with the following schedules. What value will client 1 read?

-S 01
Not deterministic, since the order of the second commands isn't given.

-S 100011
Client 1 reads 1, since client 0's second operation occurs before client 1's.

-S 011100
Client 1 reads 0, since client 1's second operation occurs before client 0's.

6 Now run with the following workload and vary the schedules as above. What happens with -S 011100? What about -S 010011? What is important in determining the final value of the file?

The final value of the file is always the same. I suspect that this is due to a quirk of the simulator and not a reflection of the file system, since the simulator always seems to increment the previous value of the file, whether the client has seen the most recent value or not.

In a more realistic simulation, the determining factor would be which client closes the file last, since last writer wins.