Homework 4

Md Ali
CS 550: Advanced Operating Systems

November 5, 2020

Exercise 1. Read Chapter 6, 7, 8

Proof. My github repo for my notes of this book is below.
https://github.com/xXxSpicyBoiiixXx/CS-550/tree/master/Textbook-Notes □

Exercise 2. What are the design issues of distributed scheduling?

Proof. I will assume that distributed scheduling is regarding to distributed algorithm within the textbook. Since this type of scheduling guarantees mutual exclusion, which only allows on process at a time access the resource. The main issue here is that it is all centralized, meaning if it crashes the whole system will go down. In addition, only having one coordination will make a potentiona bottleneck. Another issue is that there if there was a block a process can't determine what what is a permission denied or if the coordinator is dead.

□

Exercise 3. Why is message logging useful in Checkpointing? Discuss.

Proof. Messaging logging is essentially logging every message that is coming through by a process and recorded in the a so called "message log". Now the state of each of this messages are occasionally saved in a checkpoint, which is called checkpointing. This is important to note that each process is checkpointed individually, meaning no coordination is required between the checkpointing of different processes. In turn the reason this is useful is that the logged messages and checkpoints are stored in a way that survives any failures that the system is intended to recover from, meaning this segways into fault tolerance. □

Exercise 4. In fig 7.7 of the text, is signature 001001 valid for sequential consistent memory? Explain your answer.

Proof. Sequential Consistency is a data-centric consistency model if the result of any execution is the same as if the read and write operation by all processes on the data store were executed in some sequential order and the operations of each individual process appear in this quence in the order specified by its program. So going back to the problem of having 001001 as a valid signature, is not valid. This is due to the fact of the last two bytes of data. You can have a rearrangement of the first 4 bytes that are valid but when you get to the last four bytes we have already written on the fifth data byte so it will never be 0 as we have already gone through 2 write process to enable to get this far. Hence, 001001 is not a valid signature for restraints of figure 7.7. □

Exercise 5. Suppose that two processes detect the demise of the coordinator simultaneously and both decide to hold an election using the bully algorithm. What happens?

Proof. Both of these processes will get two election messages, ultimately the second election message will be ignored and the election will continue on with the pursuit of the first one. □

Exercise 6. Explain in your own words what the main reason is for considering weak consistency models.

Proof. The main reason to consider weak consistency models is that they need to have replicate a lot to gain performance. □

Exercise 7. For active replication to work in general, it is necessary that all operations be carried out in the same order at each replica. Is this ordering always necessary?

Proof. No, this ordering not always necessary. A perfect example is commutative write operations, hence with commutative properties in play, ordering will never matter. □

Exercise 8. What kind of consistency would you use to implement an electronic stock market? Explain your answer.

Proof. In the area of the stock market, the main issue here is that we need is that we need the changes to be consistent, we can utilize casual consistency in this case to satisfy this need. □

Exercise 9. Give an example where client-centric consistency can easily lead to write-write conflicts.

Proof. In the text they give an example a client A shares some data with client B and that either or of those clients modify the data but that data is stored at a different location, then this may lead to write-write conflicts. In addition, the text goes on to say that if both parties don't access the same location for a while this conflict may go on for a long time before it is discovered. □

Exercise 10. A file is replicated on 10 servers. List all the combinations of read quorum and write quorum that are permitted by the voting algorithm.

Proof. The following are all the combinations of read quorum and write quorum

(1,10)
(2,9)
(3,8)
(4,7)
(5,6)
(6,5)
(7,4)
(8,3)
(9,2)
(10,1) □

Exercise 11. Explain the difference between linearizability and sequential consistency, and why the latter is more practical to implement, in general.

Proof. Both of these gives the illusion of a single copy but the underlining difference is that linearizability cares about time while sequential consistency cares about the program order. The advantage of sequential consistency is that the system will have the freedome from different clients as long as the ordering from each client is preserver while with linearizability, all clients interleaving is pretty much determined based on time. □

Exercise 12. During the discussion of consistency models, we often referred to the contract between the software and data store. Why is such a contract needed?

Proof. Lets say a program expects a sequentially consistent data store and cannot have anything other than that. This makes the data store have to provide sequential consistency, now having this we must have a weaker model imposed hence the software must agree to the rules that given by the model. This makes the programs themselves obey rules that look like a sequentially consistent data store. □

Exercise 13. Describe a simple implementation of read-your-writes consistency for displaying Web pages that have just been updated.

Proof. There could be a mechanism in place where the browser always check whether it is displaying the most recent version of a page. This involves sending requests to a web server and is a very simple implementation of read-your-writes consistency for displaying web pages. □

Exercise 14. Is the following sequence of events allowed with a sequentially-consistent store? What about a causually-consistent data store? Explain your answer.

Proof. In the diagram, the following sequence of events are not allowed with a sequentially-consistent store. This is due to that process 3 and process 4 disagree on what will be read later down the time line. This sequence events is however allowed in causal consistent data store as all the writes happen before they are read. □