CECS 327 Assignment 6 - Coordination

20 points

Assignment Description. Answer the following questions from the Chapter 6 reading from your textbook.

Be through and complete with your answers. You may work on these questions with a partner (no

more than two working together), but both students must submit the document individually on Beachboard

Dropbox along with both students’ names on each submission.

1. Consider the behavior of two machines in a distributed system. Both have clocks that are supposed

to tick 1000 times per millisecond. One of them actually does, but the other ticks only 990 times per

millisecond. If UTC updates come in once a minute, what is the maximum clock skew that will occur?

The maximum clock skew would be the 10ms difference \* 60 secs meaning 600msec.

2. One of the modem devices that have (silently) crept into distributed systems are GPS receivers. Give

a few examples of distributed applications that can use GPS information.

Google maps is an example of an application that utilizes a distributed network of cell towers to constantly maintain connections and provide updates to the user.

3. When a node synchronizes its clock to that of another node, it is generally a good idea to take previous

measurements into account as well. Why? Also, give an example of how such past readings could be

taken into account.

It is possible that there is an error in the reading. If the average intervals are not taken into account, but they are added to the list this can occur.

4. Consider a communication layer in which messages are delivered only in the order that they were sent.

Give an example in which even this ordering is unnecessarily restrictive.

An example would be if we had to transfer a large image and made them into consecutive blocks. The person on the other end would only need to piece to blocks back together.

5. To achieve totally-ordered multicasting with Lamport timestamps, is it strictly necessary that each

message is acknowledged? Why?

No, as long as the message has a timestamp larger than the received one. When delivering to the app that other message has been received with a large timestamp guarantees that no messages come with a lower one.

6. Many distributed algorithms require the use of a coordinating process. To what extent can such

algorithms actually be considered distributed? Explain and discuss.

They can actually be considered distributed because the processes are ran on different machines despite being centralized. The coordinator forms part of the algorithm.

7. In the centralized approach to mutual exclusion (Fig. 6-15 in DS:P&P, 3rd Ed.), upon receiving a

message from a process releasing its exclusive access to the resources it was using, the coordinator

normally grants permission to the first process on the queue. Give another possible algorithm for the

coordinator and explain.

Priority requests can be handled with different level and the coordinator could just grant priority to the highest ones first.

8. Consider Fig. 6-15 again. Suppose that the coordinator crashes. Does this always bring the system

down? If not, under what circumstances does this happen? Is there any way to avoid the problem and

make the system able to tolerate coordinator crashes?

If the algorithm is set up so that every request is answered with eight a permission or denial right away. With no processes getting resources and none queued the crash would not be fatal. The next process will start a new coordinator once it sees no reply. If the coordinator saved every incoming transmission before transmitting, with a crash the new coordinator can restore that list of resources and the queue.

9. A distributed system may have multiple, independent resources. Imagine that process 0 wants to access

resource A and process 1 wants to access resource B. Can Ricart and Agrawala’s algorithm lead to

deadlocks? Explain your answer.

If the processes are gotten sequentially a process holding a resource cannot access another one and cannot block. If resource A is allowed to hold 0 and try to access B then we have deadlock. The algorithm does not lead to deadlock since each resource is handled by itself away from all others.

10. Suppose that two processes detect the demise of the coordinator simultaneously and both decide to

hold an election using the bully algorithm. What happens?

Each high numbered process will get two election messages and will ignore the second one. Then the election will continue.