Theoretical exercise for Chapter 6 - Part 2

Student name: Doan Cao Thanh Long

Student ID: 20162513

Class: ICT-02.k61

Question 1: What is a mutual exclusion algorithm in a distributed system?

Mutual exclusion algorithm in a distributed system is an algorithm to prevent race conditions. It prevents a process cannot enter its critical session while another concurrent process is currently present or executing its critical sesssion.

In distributed system, there is no shared memory or a common physical clock, so to implement this algorithm is to use the message passing approach.

Question 2: What is the drawback of the centralized algorithm for the mutual exclusion?

- If the coordinator crashes, the entire system may go down with it since it is a single point of failure.
- If processes normally block after making a request, they cannot distinguish a dead coordinator from "permission denied" since no message comes back.
- Bottleneck when deploy on the large system with a single coordinator (scalability).

Question 3: What is the drawback of the distributed algorithm for the mutual exclusion

- Algorithm is suitable only for small group of processes.
- It is highly complex due to the need of identify all processes needed.

Question 4: Propose a solution for the problem of lost token in Token Ring mutual exclusion algorithm

In Token Ring mutual exclusion algorithm, when the current token is lost, we need a mechanism to generate a new token.

Question 5:

When P3 starts the election after node P4 and P7 was broken, it will take the system a total of 10 messages to elect a new coordinator.

- 1st round, P3 sends 4 ELECTION messages to P4, P5, P6 and P7. P3 receive 2 ANSWER messages from P5, P6 -> 6 messages.
- 2nd round, P5 sends 2 ELECTION messages to P6 and P7 and receives 1 ANSWER message from P6 -> 3 messages.
- 3rd round, P6 sends 1 ELECTION message to P7 and receives 0 ANSWER message thus making it the coordinator -> 1 message.

Question 6:

a) Total number of messages needed to vote the coordinator

$$n_{messages} = 2 * (N - i - 1) + N * (i + 1)$$

with i is the index of node which detect the current coordinator is broken.

b) When a broken node become working again

- That broken node first broadcast a WORKING message to each of the other nodes for update their tables. The other working nodes will send a acknowledge message back for that previously broken nodes to update its table.
- If the said node has higher id than the current coordinator, the coordinator will sent the ELECTION message to said node, otherwise, the system will continue to run.

Question 7:

a) How many messages does the system need to successfully let a process use SR?

- Suppose there are k REQUEST messages on the queue already $k \geq 0$ (no upper limit due to Pi can post multiple REQUEST messages). Pi will wait k RELEASE messages.
- Pi sends N − 1 REQUEST messages and receives N − 1 REPLY messages.
- The total amount of messages would be

$$k + 2 * (N - 1)$$

b) The improvement will cut all the unnecessary REPLY messages since the purpose of the REPLY was to reorder the priority queues of all processes to timestamp order. The new algorithms can cut at most $N - 1$ REPLY messages. In the worst case scenario, when ts_i is the latest, it will perform like the original algorithms.