Theoretical exercise for Chapter 6 - Part 1

Student name: Doan Cao Thanh Long

Student ID: 20162513

Class: ICT-02.k61

Question 1: Give two examples to demonstrate the importance and the need of synchronization mechanism between processes in distributed systems.

Example 1: If a distributed systems is not synchronized, consider that system has 2 computer A and B in a same timezone but computer B is 15 minutes ahead of computer A. When B try to write something to the shared resources and register timestamp, the system will not let B write anything because the timestamp is invalid. In fact, users from computer B cannot write anything until the administrator synchronizes. Timestamp invalidation can also caused lagging when using real time application or application that shared the resources to a lot of users like online games or chat applications.

Example 2: Hardware and software in distributed system communicate and coordinate their actions by message parsing. For example, node 1 send a message to node 2 and 3 first, node 2 send a message to note 3 after receive message from node 1. Without time synchornization, node 3 could receive message from node 2 before the message from node 1, thus alter the ordering of events.

Question 2: Compare Network Time Protocol and Berkeley algorithm.

Network Time Protocol	Berkeley algorithm
Having 3 classes with class 1 is the highest accuracy and class 3 is the lowest	Having 1 Master to be standard
Class 1 is the standard. Class 2 get time from Class 1 and Class 2 servers. Class 3 get time from any server when they attempted connecting to one.	Master calculates average time and discards outliers.
Send timestamp to be changed into	Send time adjustment to clients
One way message to dictate time	Round trip message

Question 3: What is the typical characteristic of synchronization algorithm for wireless networks?

The typical characteristics of synchronization algorithm for wireless networks

- Carefully regiment its frequency of resynchronization, and avoid flooding
- Cannot rely on a power-hungry sources of real-time such as GPS
- A CSA in a wireless medium must continue to function in the face of node failures and recoveries
- Closely synchronize with nodes that are nearby, and more loosen with faraway nodes

Question 4: What is the difference between physical synchronization and logical synchronization

Physical synchronization	Logical synchronization
Synchronize the exact timestamp	Synchronize by order of action
Must change the clock in each systems based on a standard	No need to change the clock
Maintain the same notion of time	Keep track of information pertaining to the order of events
Expensive to maintain	Inexpensive to maintain
Inherently inaccurate	Fairly inherently accurate

Question 5: What are the update steps of counters to implement Lamport's logical clock?

- Step 1: Before executing an event Pi: `Ci = Ci + 1`
- Step 2: When process Pi sends a message m to Pj, it sets timestamp of m `ts(m)` equal to Ci
- Step 3: Upon the receipt of a message m, process Pj adjusts its local counter as Cj = max {Cj, ts(m)}, after which it then executes the first step and delivers the message to the application

Question 6:

a) The formula is not absolutely accurate, because it assumes the time sending message back and forth is equal, which is not true for every single cases

b) Consider δ be the deviation of time value and min the minimum time value it takes to transmit a message one-way. Value of δ can be determined by using 2 variables RTT and min

$$\delta = RTT - 2 * min$$

δ should be positive but it can be negative when you upgrade to a better system and/or better medium. When that happened, you need to update your min value.

Question 7:

a) Two conditions the receiving process use to check whether the message satisfies causalities

$$\begin{cases} V_{P_j}[i] = V_S[i] - 1 \\ V_{P_j}[k] \geq V_S[k] \forall k \in [1, 2, \dots, n] - i \end{cases}$$

b) Vector clock values for 4 points X1, X2, X3, and X4

$$X1 = (0, 1, 0)$$

$$X2 = (1, 1, 0)$$

$$X3 = (2, 1, 1)$$

$$X4 = (2, 1, 2)$$

c) Message will be kept at the middleware level: Message b