INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

	Stamp / Signature of the Invigilator

EXAMINATION (End Semester)									SEMESTER (Spring 2024)		
Roll Number									Section		Name
Subject Number	C	S	6	0	0	0	2	Subject Name	Distributed Systems		
Department / Center of the Student									Additional sheets		

Important Instructions and Guidelines for Students

1. You must occupy your seat as per the Examination Schedule/Sitting Plan.
2. Do not keep mobile phones or any similar electronic gadgets with you even in the switched off mode.
3. Loose papers, class notes, books or any such materials must not be in your possession, even if they are irrelevant to the subject you are taking examination.
4. Data book, codes, graph papers, relevant standard tables/charts or any other materials are allowed only when instructed by the paper-setter.
5. Use of instrument box, pencil box and non-programmable calculator is allowed during the examination. However, exchange of these items or any other papers (including question papers) is not permitted.
6. Write on both sides of the answer script and do not tear off any page. **Use last page(s) of the answer script for rough work.** Report to the invigilator if the answer script has torn or distorted page(s).
7. It is your responsibility to ensure that you have signed the Attendance Sheet. Keep your Admit Card/Identity Card on the desk for checking by the invigilator.
8. You may leave the examination hall for wash room or for drinking water for a very short period. Record your absence from the Examination Hall in the register provided. Smoking and the consumption of any kind of beverages are strictly prohibited inside the Examination Hall.
9. Do not leave the Examination Hall without submitting your answer script to the invigilator. **In any case, you are not allowed to take away the answer script with you.** After the completion of the examination, do not leave the seat until the invigilators collect all the answer scripts.
10. During the examination, either inside or outside the Examination Hall, gathering information from any kind of sources or exchanging information with others or any such attempt will be treated as '**unfair means**'. Do not adopt unfair means and do not indulge in unseemly behavior.

Violation of any of the above instructions may lead to severe punishment.

Signature of the Student

To be filled in by the examiner

Question Number	1	2	3	4	5	6	7	8	9	10	Total
Marks Obtained											

Marks obtained (in words)	Signature of the Examiner	Signature of the Scrutineer

This question booklet has 14 pages. Write the answers in the boxes only. Answers written elsewhere will not be evaluated. You may take additional sheets for rough works, but stitch them with the main answer script.

1. Answer the following questions briefly. Marks will be deducted for unnecessary descriptions. No marks will be given if the answers are not explained and only Yes/No answer is given.

   [10x3=30 Marks]

   (a) Explain whether the following statement is true of false. "*Raft supports both safety and liveness over an asynchronous system; hence, Raft violates the FLP impossibility theorem.*" Justify your answer with proper reasoning.

   (b) Consider that N number of processes are organized following a complete binary tree of height h. The root is at level 0. Assume that one of the processes at level 1 is byzantine faulty. Can we run a Byzantine fault tolerant algorithm on this process tree with the safety guarantee for the following two communication models?

      i. A process can only broadcast messages to its parent and the child processes.

      ii. A process can broadcast messages to its parent, child, and sibling processes.

   Assume that the communication model is synchronous.

(c) Assume that you have been assigned the task of developing a fault-tolerant replicated data-center for hosting a programming contest service. You know that at any point of time at most 4 of the replicas can crash. Among the remaining replicas, at most 3 replicas are vulnerable to a security attack. You want to ensure that the service is both crash as well as Byzantine fault-tolerant. How many total number of replicas do you want to place at minimum to ensure this? Assume that the service uses an asynchronous environment for computing and communication.

(d) What is meant by a *stable checkpoint* in PBFT? What is the significance of stable checkpoints in the context of ensuring safety in PBFT?

(e) Assume that you developed a protocol to ensure total order delivery of messages between a set of processes over a synchronous, reliable, pair-wise FIFO communication channel. Does it ensure (a) Linerizability, (b) Sequential Consistency, and (c) Causal Consistency? Explain your answers briefly.

(f) Assume that you have used vector clock in a synchronous, reliable communication system where a set of processes exchange messages among each other to trigger events. The events are ordered in a partial order of the vector clock readings. Does this system ensure causal consistency?

(g) Does the order of message delivery create any impact on CmRDT? Explain with an example.

(h) Assume that there are two processes P and Q. P has three send events with their vector clock timestamps as S1 ($< 1,0 >$), S2 ($< 2,0 >$), and S3 ($< 3,0 >$), and Q has three receive events with their vector clock timestamps as R1 ($< 3,1 >$), R2 ($< 3,2 >$), and R3 ($< 3,3 >$). The first entry in the vector clock corresponds to the events in P, and the second entry corresponds to the events in Q. Can you uniquely associate the send events with the corresponding receive events (which receive event corresponds to which send event) by analyzing the vector clock values? If you can, then show the association and explain why it is unique; otherwise, show a counter example indicating that multiple associations are possible.

(i) Consider three processes P_1, P_2 and P_3 who are communicating over asynchronous, unidirectional, reliable, FIFO channels. Say, P_1 broadcasts message M. Once the other processes receive the broadcast message, they send back a reply R_M to P_1. Assume that there would not be any failure during the communications. Draw the lattice corresponding to the runs in this system. How many different runs will be possible? Consider Lamport's clock to order the messages.

(j) Consider that two players P_1 and P_2 are playing the Counter-Strike (CS) game, where the individual CS servers are hosted on their local machines. Say, P_1 is trying to attack on P_2, and P_2 is trying to escape. When P_1 makes a move (say, attack by firing on P_2), the state of P_1 is sent to P_2 through a message passing, and vice-versa. Assume that the communication channel between P_1 and P_2 is reliable and FIFO but asynchronous. Can you use a vector clock to synchronize the moves of P_1 and P_2 in this case so that the game runs correctly? By synchronization, we mean that P_2 should not get killed by P_1 because the state of P_2 has not been updated to P_1 when P_1 attacks P_2. Explain your answer.

2. (a) Consider four replicas R_1, R_2, R_3, and R_4 where R_1 is the primary and others are the backups. Consider that R_1 receives a message M from a client C and runs PBFT to commit the message under asynchronous communication channels with a maximum of one failure. Assume that R_3 fails to send the PREPARE message to R_2. Use a vector clock to mark the clock values for all the *send* and the *receive* events for the messages exchanged by the replicas, and show that the *send* event of the COMMIT message at any one of the replicas ***causally succeeds*** the *send* event of the PREPARE messages across majority of the replicas. **[8 Marks]**

(b) The *total order delivery* of two messages M1 and M2 is defined as follows. If a process delivers message M1 followed by M2, then all processes delivering both M1 and M2 must deliver M1 first followed by M2. Note that total order delivery guarantee is important for many distributed protocols, such as the consensus messages need to be totally ordered. Can you use vector clocks to ensure total order delivery? Explain your answer with a proof or show a counter example to disprove. **[7 Marks]**

3. (a) Consider that you have a five server Raft system. Suppose the state of the servers' logs are as follows, where the notation N(T) means N^th log entry from the term T.

```
S1:  4(1)  5(1)
S2:  4(1)  5(2)
S3:  4(1)  6(1)
S4:  4(1)  6(1)  7(2)
S5:  4(1)
```

Consider that the leader at term 2 fails and the servers are about to choose a new leader.

Can any replica have already executed the operation in log entry 6 before the Term 3 starts? Explain your answer. **[3 Marks]**

Could operation 5(2) be committed in the future after the leader of Term 3 starts? Assume that the client does not rebroadcast the operation. Explain your answer. **[3 Marks]**

Could operation 5(1) be committed in the future after the leader of Term 3 starts? Assume that the client does not rebroadcast the operation. Explain your answer. **[3 Marks]**

(b) During the last year classes of this course, one of students came up with an idea that we can make Raft faster by processing the read-only operations as follows. As read-only operations do not involve any update in the state machine, the leader can execute them locally and send back the reply directly to the clients, without replicating the operation to the followers. Do you think that this idea will work good in practice? Explain your answer. **[3 Marks]**

(c) Suppose a Paxos cluster of eight acceptors along with a bunch of clients gets partitioned in half due to network failures. Some clients end up communicating with one set of acceptors while others communicate with the remaining acceptors. Four acceptors agree to one value while four others agree to a different value. I argue that the consensus is not achieved as the system agreed upon two different values. Is there anything wrong in this argument? Explain your answer.
[3 Marks]

4. (a) What is the purpose of the *prepare* phase in PBFT? Give an concrete example where PBFT will fail if the *prepare* phase is not there. To draw the example, consider a scenario where the replicas forward a *commit* message directly if they agree upon the received *pre-prepare* message. Now once a replica receives $2f+1$ *commit* messages, then they forward the *response* back to the client. Draw a scenario and show that this modified version of PBFT may fail to satisfy the consensus safety. **[5 Marks]**

(b) Assume a PBFT system with four replicas, and one of them is the primary. What is the minimum number of faulty replicas that an attacker needs to make the system inconsistent; i.e. different non-faulty replicas will execute requests in different orders? With these minimum number of faulty replicas, show an example of how the system may behave inconsistently under an attack. **[6 Marks]**

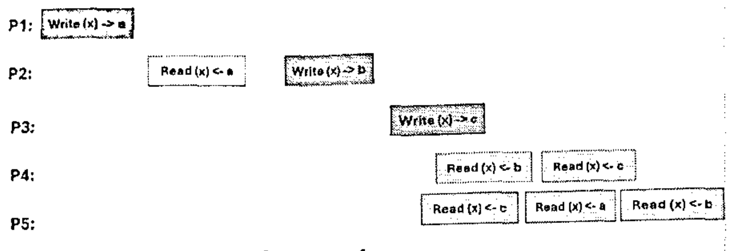
(c) Consider that four processes P1, P2, P3 and P4 run a voting protocol to ensure sequential consistency. Assume that the read and write quorums for the processes are as follows.

Process	**Read Quorum**	**Write Quorum**
P1	{P1, P3, P4}	{P1, P3, P4}
P2	{P2, P3 }	{P3, P4}
P3	{P3, P4}	{P3, P4}
P4	{P1, P4}	{P1, P3, P4}

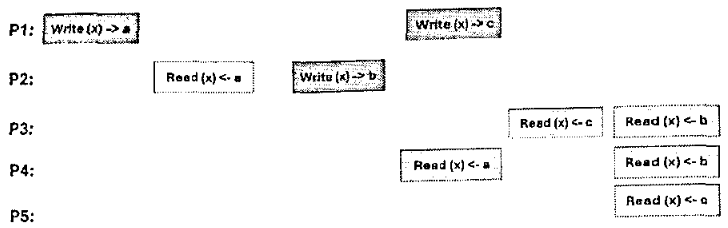
Check whether the processes will be able to run the voting protocol correctly. If yes, explain your answer. If now, show an example where a conflict might arise.

[4 Marks]

5. (a) Consider the following two sequences of operations over multiple processes, as shown as Sequence 1 and Sequence 2. P1 to P5 are the five different processes.



Sequence 1



Sequence 2

Explain whether these two sequences are (a) Sequentially Consistent, (b) Causally Consistent? Explain you answer for both the cases. **[6 Marks]**

```
Sequence 1:
    Sequentially Consistent? YES / NO
    Causally Consistent? YES / NO
    Explanation:







Sequence 2:
    Sequentially Consistent? YES / NO
    Causally Consistent? YES / NO
    Explanation:
```

(b) Consider that there are four processes P1, P2, P3 and P4, which collectively process a graph G. The processes can either add an edge to the graph through the `addEdge(G)` operation, or can remove an edge from the graph through the `removeEdge(G)` operation. However, there is a constraint that the graph must be a DAG (Directed Acyclic Graph) eventually after the the operations are processed.

   i. Is it possible to use a CvRDT to ensure eventual consistency for the above operations?

   ii. If the graph is a monotonic graph, that is only the `addEdge(G)` operation is possible and no `deleteEdge(G)` operation is permissible, then can you use a CvRDT to ensure eventual consistency?

   Explain your answer for both the cases.

[9 Marks]