CS60002: Distributed Systema
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur
Midsem Examination, Spring 2014

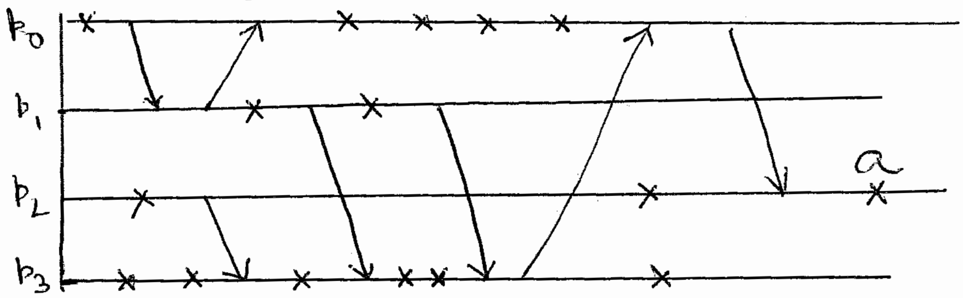
Total Marks: 100 **Time: 2 Hours**

INSTRUCTIONS: Please read carefully before starting

1. *Answer ALL Questions.*
2. *Unless otherwise mentioned, assume that systems are asynchronous with distinct node ids, and all communications are reliable.*
3. *Write clearly at the beginning of your answer any additional assumptions that you make which are not given in the question, though making unnecessary assumptions may incur a penalty.*

---

1. **(a)** With respect to the space-time diagram given below (x's indicate local computation events), what would be the Lamport's logical timestamp and vector clock timestamp of the event *a*? (5)



(b) Let *a*, *b*, *c* be three events in a distributed system such that each of them belongs to a different process. Using Lamport's definitions of causal ordering of events, state whether the following statements are true or false:

a. if *a* and *b* are concurrent, and *b* happened before *c*, then *a* happened before *c*
b. if *a* and *b* are concurrent and *b* and *c* are concurrent, then *a* and *c* are concurrent

Justify your answers briefly. (4)

(c) Define a consistent global state. When is a global state said to be transitless? (5)

(d) Show clearly with an example involving two processes only (i) why Lamport's mutual exclusion algorithm requires FIFO channels and (ii) why Raicairol-Carvalho's mutual exclusion algorithm does not ensure timestamp-ordered entry into the critical section. Just give the space-time diagram and a 1-2 sentence description, do not write long explanations. (6)

(e) Consider two clocks, one with drift rate $\delta 1$ and the other with drift rate $\delta 2$. If the clocks are resynchronized every ρ seconds, and they can be synchronized to within θ of each other, find an expression for the maximum skew of the clocks at the beginning of each resynchronization interval. (5)

2. **(a)** Suppose you are given a connected, undirected graph with a special node *i* and an upper bound *T* on the message delay over any link. Design a global state collection algorithm that does not rely on FIFO channels and does not use piggybacking. At the end of your algorithm, the global state should be available at node *i* and *i* should know that state collection is complete.

Analyze its message complexity. Clocks in the nodes are not synchronized, but you can assume that they have no drift. Assume that every process knows an upper bound on the no. of processes in the system, and processing times at nodes are negligible. (20)

(b) Given an arbitrary synchronous network with unique ids, give a protocol that elects exactly k leaders in the network (k is known to all nodes). When your protocol ends, exactly k nodes will declare themselves as leaders, and all other nodes will know that they are not leaders. Analyze the time and communication complexity of your algorithm. (10)

(c) Consider implementing Raymond's algorithm in a system where the token message and the request message each consist of 32 bits. The links are fast; but the nodes are slow. So it is more important to reduce the no. of messages rather than the size of a message, as long as the message size is constant (i.e., independent of the network size). Suggest a modification to reduce the message complexity of Raymond's algorithm under heavy load in this system (analyze to give an idea of the amount of reduction achieved). Is the strategy also helpful under light load? Justify. (10)

3. **(a)** Design an asynchronous distributed algorithm to find the shortest path from a designated node X to all other nodes in an undirected, connected graph with all edge weights equal to 1. X will be the only node to initiate the algorithm, and no flooding should be used. At the end of the algorithm, each node should have a local variable value set to its shortest path length from X. The exact path need not be maintained. What is the message complexity of the algorithm? Do not worry about how X gets to know when all the values are correctly computed at the nodes (i.e., about detecting termination of the algorithm at X). (20)

   (b) Suppose you are given a distributed algorithm to find an unrooted spanning tree (so there is no parent or child pointers, every node only knows which of its edges belong to the spanning tree) of a connected network. How can you use this algorithm to elect a single leader node in the network? At the end of your algorithm, only one process will declare itself the leader (other processes may or may not know who the final leader is). What is the message complexity of your algorithm? Note that you do NOT have to design the algorithm to find the unrooted spanning tree, it is given to you and you have to use it to design your leader election algorithm. (15)