Homework 2 (Multicast, Mutual Exclusion, Leader Election) - 100 Points

CS425/ECE 428 Distributed Systems, Fall 2009, Instructor: Klara Nahrstedt

Out: Tuesday, September 22, **Due Date**: Tuesday, October 6

Instructions: (1) Please, hand in hardcopy solutions that are typed (you may use your favorite word processor). We will not accept handwritten solutions. Figures and equations (if any) may be drawn by hand. (2) Please, start each problem on a fresh sheet and type your name at the top of each sheet. (3) Homework will be due at the **beginning of class** on the day of the deadline.

Relevant Reading for this Homework: Chapter 12.1-12.4

Problem 1: Ordered Multicast – 20 Points

Consider the Figure 1. Using sequence numbers (for FIFO Ordering multicast) or vector clocks (for Causal Ordering multicast) , mark states at the point of each multicast send and each multicast receipt. Also mark multicast receipts that are buffered, along with the points at which they are delivered to the application

a. (10 Points) FIFO Ordering multicast algorithm as discussed in class

b. (10 Points) Causally Ordered multicast algorithm as discussed in class

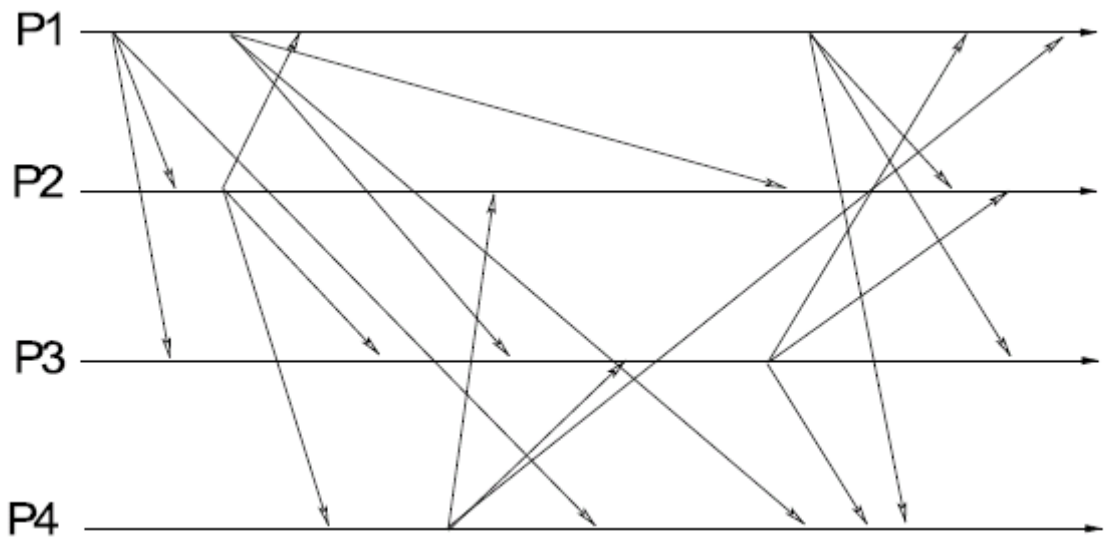

Figure 1: *Ordered Multicast Problem*

Problem 2: Ordered Multicast - 20 Points

2.1 (**10 Points**) Let us assume FIFO-ordered multicast per group, i.e., if a correct process issues **multicast(g,m)** and then **multicast (g, m')** , then every correct process that delivers m' will have already delivered m. Show that the *FIFO-ordered multicast algorithm per group* does not work for overlapping groups, by considering two messages sent from the same source to two overlapping groups, and considering a process in the intersection of those groups. Adapt the protocol to work for this case and write down the algorithm. Hint: processes should include with their messages the latest sequence numbers of messages sent to *all groups.*

2.2 (**10 Points**) Let us assume *FIFO-ordered multicast across groups*, i.e., (a) FIFO-ordered multicast per group, and (b) if a correct process issues **multicast(g,m)** and then multicast(g'm'), then every correct process in the intersection of groups g and g' that delivers m' will have already delivered m. Does the *FIFO-ordered multicast algorithm across groups* work for overlapping groups, when considering two messages sent from the same source to two overlapping groups and considering a process in the intersection of those groups? Explain if yes or no. In case it does not work, adapt the protocol to work for this case and write down the algorithm.

Problem 3: Distributed Mutual Exclusion – 20 Points

The following Figure 2 shows a Raymond's Ring for management of mutual exclusion. Node 1 is holding the token (is in CS). Current queue entries are shown under each node.

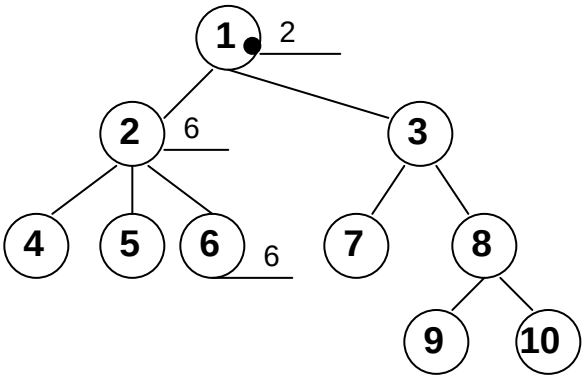

Figure 2: *Raymond Algorithm – Initial State from which you start*

a. Suppose nodes 9, 7, and 5 (in that order) request to enter the same CS while node 1 is still holding the token. List contents of each node's queue, after these requests are processed by all relevant nodes.
b. Assuming no other requests are received for entry to the same CS, other than those in part (a), show the contents of each node's queue when the token gets to node #6
c. Assuming no other requests are received for entry to the same CS, other than those in part (a), show the contents of each node's queue when the token gets to node #9.
d. Consider a request from node 10 while node 9 is holding the token, show the contents of each node's queue after this request is processed by all relevant nodes.

Problem 4: Distributed Mutual Exclusion – 10 Points

Consider a group of distributed systems, P1, P2, P3, and P4 that share an object. They use the Ricarti-Agrawala algorithm for management of mutual exclusion. P1 is currently in the critical section and there is no other node in the "wanted" state. Now consider requests from P4, P2 and P3 (in that order) to enter the same CS and assume that these request messages from P4, P2, P3 arrive at other peers in the order of requests.

a. **(5 Points)** Show the state (as required by the algorithm, i.e. "held", "wanted", etc.) and queue entries at each processor.
b. **(5 Points)** Now, P1 exits the CS and informs all relevant nodes that CS is released. Show the state and queue entries at each processor, at this stage.

Problem 5: Election Algorithms – 10 Points

Consider a group of distributed systems, P1, P2, P3, P4 and P5 (P5 is currently the coordinator). P5 fails and P2 notices the failure. If the Bully algorithm is used for election of a new coordinator and the election attribute is the (Max of) processor numbers, show the set of all messages communicated through each communication channel P_{ij} i,j = 1..5 for this election. Show the type of each message as "election", "response", "coordinator".

Problem 6: Election Algorithm – 20 Points

For a synchronous system, design an election algorithm using a mutual exclusion algorithm, without knowing how the mutual exclusion algorithm is implemented.