CS454/654 Assignment 4
Winter 2016

Due: 28 March 2016 (4:00 pm)*
Returned: 11 April 2016
Appeal deadline: 18 April 2016

Question 1 (20%) In Lamport's logical clocks (where such a clock for an event v is denoted by $C(v)$), by considering a chain of zero or more messages connecting events e and e' and using induction show that

$$e \rightarrow e' \Rightarrow C(e) < C(e')$$

Question 2 (30%) Consider two transactions T_i and T_j which both transfer funds from an account x located at site 1 to an account y located at site 2. As we are interested only in the read and write operations, the two transactions can be described by the following sequence of operations

T_i : $R_i(x)$ $W_i(x)$ $R_i(y)$ $W_i(y)$
T_j : $R_j(x)$ $W_j(x)$ $R_j(y)$ $W_j(y)$

assuming that each transaction reads x, decrements this value of x it has read, writes the new value, reads y, increments this value of y it has read, and writes the new value.
Assume that these two transactions are activated almost simultaneously and consider the following execution at each site:

site 1	site 2
$R_i(x)$	
$W_i(x)$	
$R_j(x)$	$R_i(y)$
$W_j(x)$	$W_i(y)$
	$R_j(y)$
	$W_j(y)$

In this representation of execution, the fact that operation $R_i(y)$ appears next to $R_j(x)$ means that these two operations start at the same time. Therefore, T_j begins reading x immediately after T_i has written x, although T_i is not yet terminated.

1. Is this distributed execution schedule serializable? If so, explain briefly how you can infer that it is serializable, and what is the global serialization order. If it is not serializable, why not?
2. Can this distributed execution schedule be generated by a strict 2-phase locking protocol (I don't care if it is a centralized or a distributed 2PL mechanism)? If it can, indicate the locking/unlocking sequence. If it cannot, explain why not.

*Please see the Evaluation section under the General Course Information web page for late submission policy.

3. Can this execution be produced by the basic timestamp mechanism? If it can, indicate what the timestamp values and conditions need to be. If it cannot, explain why not.

Question 3 (20%) Consider a distributed system in which data item x is replicated at sites S_1, S_2 and S_3. Site S_1 is local to client C while the other sites are remote to C. Design a valid majority quorum system that will support consistent reading and writing of data item x for all clients while providing the best performance to client C for executing reads and writes on x (ensure that you justify how your majority quorum system design achieves this).

Question 4 (10%) Consider three processes executing the statements shown below. Assume that a, x, y, and z are initially 0. What are the possible values for a that can result from a sequentially consistent interleaving of the statements of the three processes? For each value of a, show one sequentially consistent interleaving yielding that value.

Process A	Process B	Process C
$A_1: \ x = 1$ $A_2 : if \ (y == 0) \ a = a + 1;$	$B_1: \ y = 1$ $B_2 : if \ (z == 0) \ a = a + 1;$	$C_1: \ z = 1$ $C_2 : if \ (x == 0) \ a = a + 1;$

Question 5 (20%) Consider the following failure scenarios in a distributed system that uses 2PC among the coordinator component and participant components:

1. the coordinator fails after it sends the *prepare* message to participants and before it receives the *prepared* message from all participants.
2. a participant fails after sending the *prepared* message to the coordinator and before it receives the *global-commit* or *global-abort* message from the coordinator.

Assume that:

- the coordinator always processes at least a portion of a global transaction.
- messages are never lost, and that if either the coordinator or a participant waits for messages for a period of time longer than a set *timeout* value, it will conclude that the component from which it is waiting for a message has failed.
- in each of the scenarios above, the timeout value is exceeded before the failed component recovers.

For each of the two scenarios above, describe the actions of the component upon recovery, justifying the action you describe.