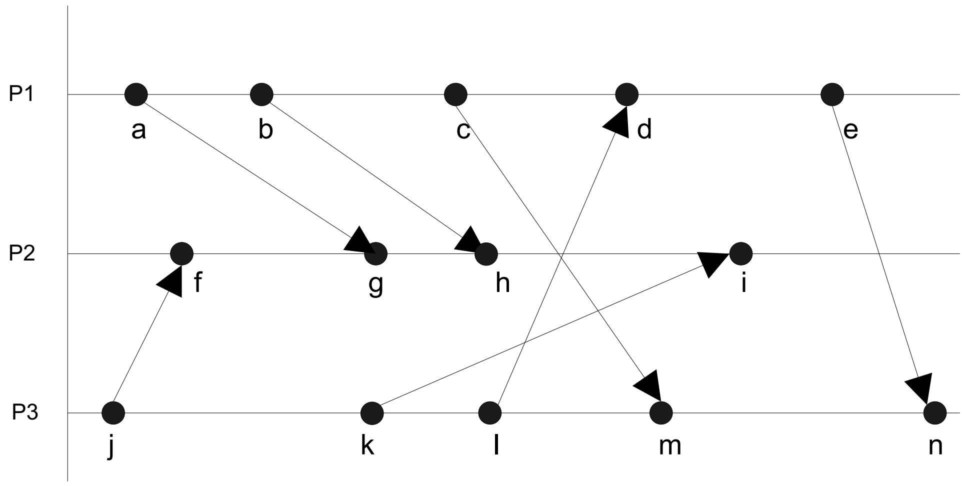
ECLT5820 Distributed and Mobile Systems

Assignment 2 (Topics 5-8)

Due – 11:59pm, 4th Nov., 2020 (Wednesday)

Note: Please send a pdf file to eclt5820@cse.cuhk.edu.hk with email title and file name "ECLT5820 Asg#2, Your name, Your student ID".

Q1 (**20 points**) Please draw **Lamport's Timestamp** and **Vector Clock** for the following operations (assuming P1, P2 and P3 all start at 0).



Q2 (**20 points**) Explain why serial equivalence requires that once a transaction has released a lock on an object, it is not allowed to obtain any more locks.

A server messages objects $a_1, a_2, \ldots, a_n$. The server provides two operations for its clients:

read (i) returns the value of a_i*;*

write(i, value) assigns value to a_i*.*

The transactions T and U are defined as follows:

T: x= read (i); write(j, 44);

U: write(i, 55); write(j, 66);

Initial values of a_i and a_j are 10 and 20, respectively.

(1) Describe an interleaving of the transactions T and U in which locks are released early with the effect that the interleaving is not serially equivalent. Please give an interleaving of T and U, and show what would happen if locks are released too early.

(2) Which of the following interleavings is serially equivalent and which could occur with two-phase locking?

(a)

T	*U*
x=read(i);	
	write(i, 55);
write(j, 44);	
	write(j, 66);

(b)

T	*U*
x=read(i); *write(j, 44);*	
	write(i, 55); *write(j, 66);*

(c)

T	*U*
	write(i, 55);
	write(j, 66);
x=read(i);	
write(j, 44);	

(d)

T	*U*
	write(i, 55);
x=read(i);	
	write(j, 66);
write(j, 44);	

Q3 (**20 points**) A conflict exists when two transactions access the same item, and at least one of the accesses is a write. The followings are three interleaving schedules:

Schedule 1	
T1	T2
read(B)	
B=B+10	
write(B)	
	read(A)
	t=A*0.2
	A=A-t
	write(A)
	read(B)
	B=B+t
	write(b)
read(A)	
A=A+20	
write(A)	

Schedule 2	
T1	T2
read(B)	
B=B+10	
write(B)	
read(A)	
A=A+20	
write(A)	
	read(A)
	t=A*0.2
	A=A-t
	write(A)
	read(B)
	B=B+t
	write(B)

Schedule 3	
T1	T2
read(B)	
B=B+10	
write(B)	
	read(A)
	t=A*0.2
	A=A-t
	write(A)
read(A)	
A=A+20	
write(A)	
	read(B)
	B=B+t
	write(B)

(1) Are these schedules serially equivalent interleavings? If so, what are they serially equivalent to? Please explain.

(2) Assuming transactions T1 and T2 will commit only after the last operation (i.e., write(A) for T1 and write(B) for T2). Which transaction(s) are strict transaction(s)? Identify which of the following problems is encountered in any of the three schedules (please indicate all such problems in these transactions, if any):
   a) dirty reads problem
   b) premature writes problem (assuming 'overlaps' are not allowed for write operations of different transactions)

   Please explain your answers in detail.

Q4 (**20 points**) A server messages objects $a_1, a_2, \ldots, a_n$. The server provides two operations for its clients:

read (i) returns the value of a_i*;*

write(i, value) assigns value to a_i*.*

Assuming all values are initialized to 0. The transactions T and U are defined as follows:

T: x= read (i); y = read (j); write(i, 33); write(j, 44);
U: x= read(k); write(k, 66); write(i, 55); y = read (j); write(j, 66).

(1) For the serially equivalent interleaings of the transactions T and U, what are the possible results? What will be the values of x, y, i, j, and k in these results?
(2) Give two serially equivalent interleavings of the transactions T and U.

Q5 (**10 points**) In a decentralized variant of the two-phase commit protocol the participants communicate directly with one another instead of indirectly via the coordinator. In Phase 1, the coordinator sends its vote to all the participants. In Phase 2, if the coordinator's vote is No, the participants just abort the transaction; if it is Yes, each participant sends its vote to the coordinator and the other participants, each of which decides on the outcome according to the vote and carries it out. Calculate the number of messages and the number of rounds it takes. What are its advantages or disadvantages in comparison with the centralized variant?

Q6 (**10 points**) A server manages objects $a_1, a_2, \ldots, a_n$. The server provides two operations for its clients:

read (i) returns the value of a_i*;*

write(i, value) assigns value to a_i .

The transactions *T*, *U* and *V* are defined as follows:

T: x= read (i); write(j, 44);
U: write(i, 55); write(j, 66);
V: write(k, 77); write(k, 88);

Describe the information written to the log file on behalf of these three transactions if strict two-phase locking is in use and *U* acquires a_i and a_j before *T*. Describe how the recovery manager would use this information to recover the effects of *T*, *U* and *V* when the server is replaced after a crash. What is the significance of the order of the commit entries in the log file?