

## CS143: Homework 5

**Problem A:** T1 is the oldest transaction, and T3 is the youngest.

T1	T2	T3
write D		
	write C	
read C		
		read C
		write A
	read A	
write A		

1. Is this first schedule conflict-serializable<sup>1</sup>?

Now assume that the transaction manager uses a 2PL protocol where each exclusive/shared lock is set just before it is needed for the write/read action.

2. If we do not use any deadlock prevention strategy, will the resulting transactions (i) complete, or (ii) deadlock<sup>1</sup>? If your answer is (i), show a completed schedule; if it is (ii) show the schedule up to the deadlock.
3. If we use a wait-die deadlock prevention strategy will the resulting transactions (i) complete, or (ii) deadlock<sup>1</sup>? If your answer is (i), show a completed schedule; if it is (ii) show the schedule up to the deadlock.

Now consider a second schedule, as follows:

T1	T2	T3
write D		
	write C	
read C		
		write A
		read C
	read A	
write A		

4. Is this second schedule conflict-serializable<sup>1</sup>?

Now assume that the transaction manager uses a 2PL protocol where each exclusive/shared lock is set just before it is needed for the write/read action, and answer the following questions for this second schedule.

5. If we do not use any deadlock prevention strategy, will the resulting transactions (i) complete, or (ii) deadlock<sup>1</sup>? If your answer is (i), show a completed schedule; if it is (ii) show the schedule up to the deadlock.
6. If we use a wound-wait deadlock prevention strategy will the resulting transactions (i) complete, or (ii) deadlock<sup>1</sup>? If your answer is (i), show a completed schedule; if it is (ii) show the schedule up to the deadlock.

---

<sup>1</sup>Justify your answer using the applicable graph

**Problem B** 1. Consider the following schedule: (w3(A) means that transaction T3 writes A, C3: T3 commits):

w3(A) r1(A) c3 w1(B) c1 r2(B) w2(C) r4(B) c2c4

- (a) Is it a serial schedule?
- (b) Is the schedule conflict serializable? If so, what are all the equivalent serial schedules?
- (c) Is the schedule recoverable? If not, can we make it recoverable by moving a single commit operation to a different position?
- (d) Is the schedule cascadeless? If not, can we make it cascadeless by moving a single commit operation to a different position?