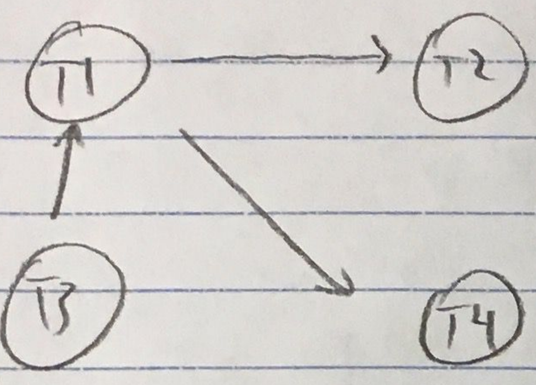
Jorm Lees
409-640-158

CS143 Homework 5(cont)

Prob B (1) (a) [No]

- As T1 reads A before T3 has committed

(b)



[yes]

w3(A) c3 r1(A) w1(B) c1 w2(C) c2 r4(B) c4
w3(A) c3 r1(A) w1(B) c1 r4(B) c4 w2(C) c2
w3(A) c3 w1(B) r1(A) c1 w2(C) c2 r4(B) c4
w3(A) c3 w1(B) r1(A) c1 r4(B) c4 w2(C) c2

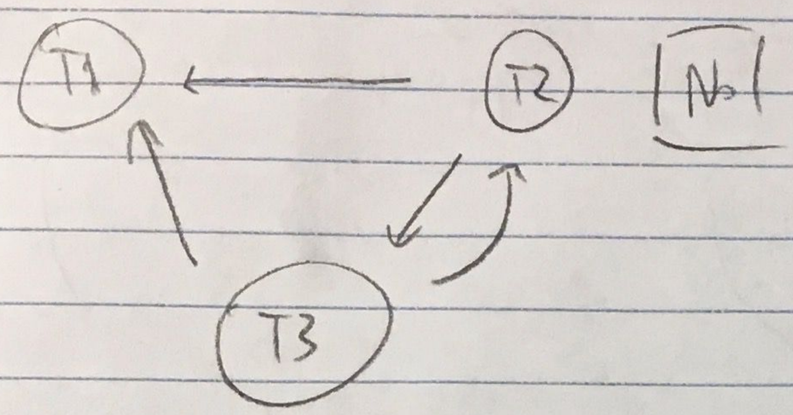
(c) [Yes]

- As commits of transactions who write to a data item Q appear before commits of transactions who read from Q

(d) [No]

- If c3 goes before r1(A), then it is cascadeless

④



⑤

T1	T2	T3
lock-X(B)		
write B		
	lock-X(C)	
	write C	
wait for C		
		lock-X(A)
		write A
		wait for C
	wait for A	

(ii) Re-clock

• for same reason described in ⑤

⑥

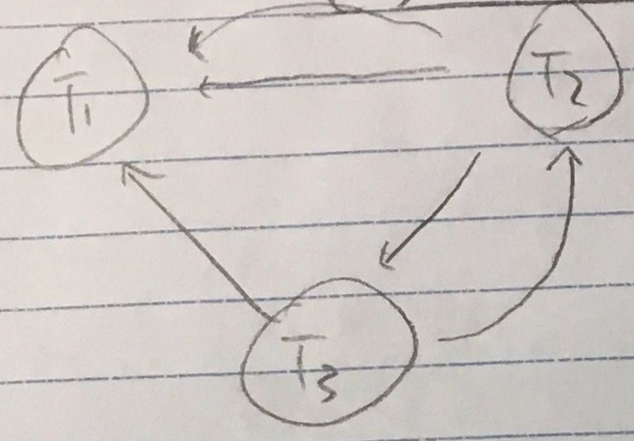
T1	T2	T3
lock-X(B)		
write B		
	lock-X(C)	
	write C	
	rollback T2	
lock-S(C)		
read C		
		lock-X(A)
		write A
		wait for C
		rollback T3
lock-X(A)		
write A		
unlock B		
unlock C		
unlock A		

(i) Complete

Jason Lee
404-640-158
CS143, Zaniolo

CS143 Homework #5

Prob A | 1



No 1

(2)

T1	T2	T3
lock-X(D)		
write D		
	lock-X(C)	
	write C	
wait for C		
	wait for C	lock-X(A)
		write A
	wait for A	
wait for A		

(ii) Deadlock

- As Basic 2PL has a growing phase followed by a shrinking phase, T2 can't unlock C until it locks A, thus it holds lock C. In the meantime, T3 is waiting for lock C, and locks A. T3 can't unlock A, until it acquires lock C, so deadlock

(3)

T1	T2	T3
lock-X(D)		
write D		
	lock-X(C)	
	write C	
wait for C		
		rollback T3
	lock-S(A)	
	read A	
	unlock C	
	unlock A	
lock-S(C)		
read C		
lock-X(A)		
write A		
unlock C		
unlock A		

(i) Complete