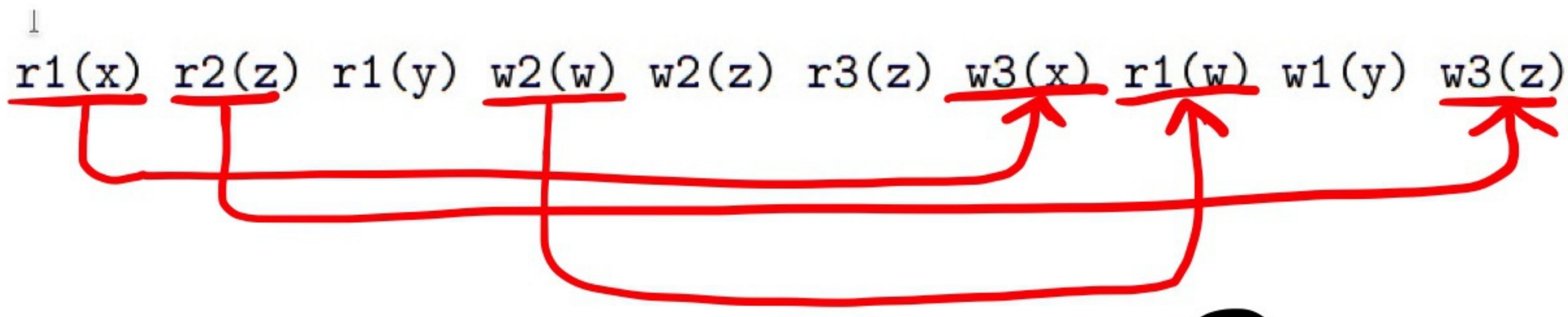
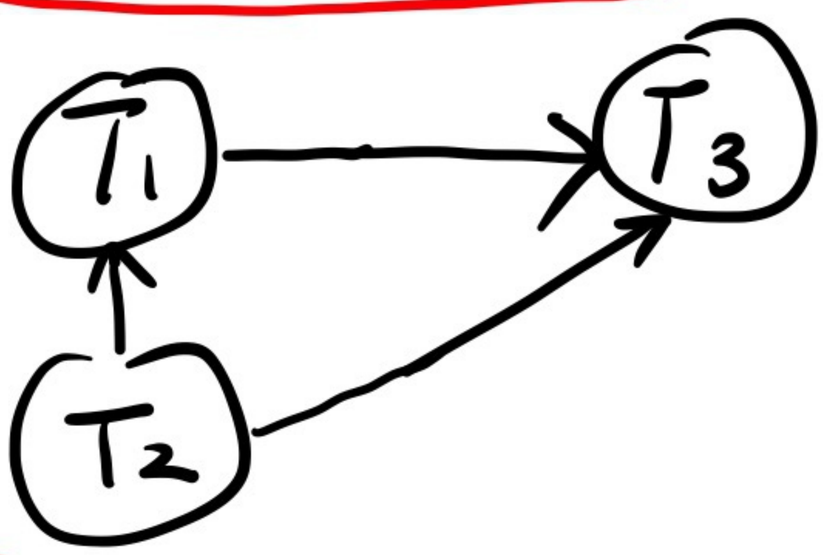
Q2

S1: r1(x) r2(z) r1(y) w2(w) w2(z) r3(z) w3(x) r1(w) w1(y) w3(z)



a) Conflict Graph:



∴ Serializable

b)

S1: r1(x) r2(z) r1(y) w2(w) w2(z) r3(z) w3(x) r1(w) w1(y) w3(z)

T_1 : unable to acquire lock.

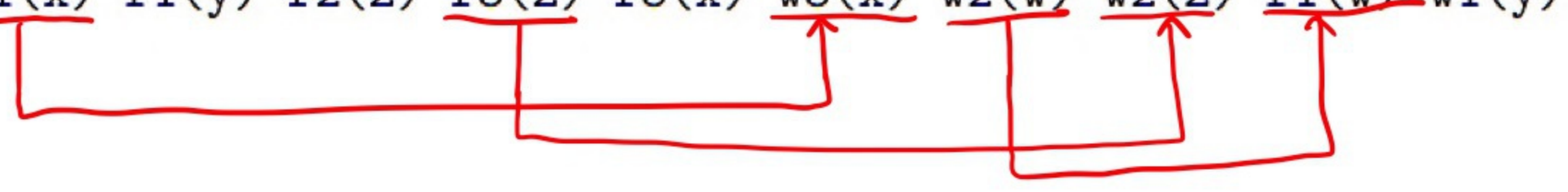
$SL_1(x)$ $SL_2(z)$ $SL_1(y)$ $XL_2(w)$ $XL_2(z)$ $SL_3(z)$ [$UL_1(x)$, $XL_3(x)$]

T_1 shrinking

∴ Not possible to obtain schedule, T_1 will try acquiring SL on W in shrinking phase.

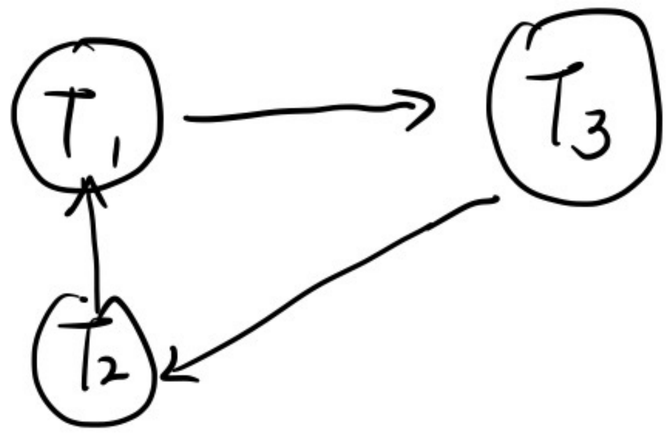
(2)

S2: r1(x) r1(y) r2(z) r3(z) r3(x) w3(x) w2(w) w2(z) r1(w) w1(y)



a)

Conflict Graph :



∴ NOT serializable

b) Not possible, since this schedule is not serializable, there will always a cycle prevent valid schedule.

Q3

LOG:

LSN	Entry
100	T1 update P2
101	T2 update P1
102	T2 commit
103	T3 update P1
104	T3 update P3
105	T1 update P4
106	T3 commit

Data pages:

Data Page	LSN of Last recorded log entry
P1	101
P2	100
P3	104
P4	-

a)

P_2 100

P_3 104

P_1 ~~101~~ 103

P_4 105 →.

T_1 aborted

T_4 should be aborted.

1. First, redo : 103
2. Then, undo: 100

b)

No Force, since T_3 is partically written to data page after commie, which won't happen if force used

c) STEAL used, as T_1 is an uncommited transaction but its changes refleced to the DATA page., also, no force is used, which makes it impossible to be part of the force action.