# Transaction Problems

**Note: The notion RT1​ (A) and WT1​ (A) means T1​ reads and writes object A respectively.**

1. ​Draw the precedence graph for the schedule

W​ T1(A) WT2​ (A) RT3 (B) WT3​ (B) RT1​ (B)

T3->T1->T2

2. For each of the following schedules of reads and writes done by transactions, indicate whether the schedule is conflict serializable, or serial:

1) RT1​ (A) RT2​ (A) WT1​ (A) RT2​ (B)

Conflict serializable

2) RT1(A) WT1 (A) RT2(A) W​T2 (B)

Conflict serializable; Serial

3) WT1​ (A) WT2​ (A) W​T1 (B) WT2​ (B) WT1​ (B)

None

3. Consider the following two transactions and schedule

RT1(A) WT1(A) RT2(A) RT2(B) RT1(B) WT1(B)

1) is this schedule conflict-serializable? Explain why or why not.

No, because the precedence graph is as below, which is not acyclic. A schedule is conflict serializable if and only if its precedence graph is acyclic.

A

B

2) Show how 2PL can ensure a conflict-serializable schedule for the same transactions above. (use 2PL to re-reschedule the transactions)

|  |  |  |
| --- | --- | --- |
| Time | Transaction T1 | Transaction T2 |
| T1 | XLOCK A |  |
| T2 | Read A |  |
| T3 | Write A | SLOCK A |
| T4 | XLOCK B | WAIT |
| T5 | Read B | WAIT |
| T6 | Write B | WAIT |
| T7 | UNLOCK X(A) | Read A |
| T8 | UNLOCK X(B) | SLOCK B |
| T9 | COMMIT | Read B |
| T10 |  | UNLOCK S(A) |
| T11 |  | UNLOCK S(B) |
| T12 |  | COMMIT |