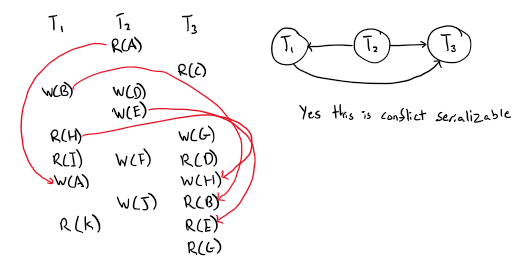
HW7

1. Consider the schedule given below in Table below. R(·) and W(·) stand for ‘Read’ and ‘Write’, respectively. T stands for transactions and t stands for time stamps.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | t1 | t2 | t3 | t4 | t5 | t6 | t7 | t8 | t9 | t10 |
| T1 |  |  | W(B) |  | R(H) | R(I) | W(A) |  | R(K) |  |
| T2 | R(A) |  | W(D) | W(E) |  | W(F) |  | W(J) |  |  |
| T3 |  | R(C) |  |  | W(G) | R(D) | W(H) | R(B) | R(E) | R(G) |

1. Give the dependency graph of this schedule.
2. Is this schedule conflict serializable? If you answer “yes”, provide the equivalent serial schedule. If you answer “no”, briefly explain why.
3. Is this schedule possible under 2PL?



**Equivalent Serial Schedule T2->T1->T3:**

**R2(A), W2(D), W2(E), W2(F), W2(J), W1(B), R1(H), R1(I), W1(A), R1(K), R3(C), W3(G), R3(D), W3(H), R3(B), R3(E), R3(G)**

**Yes, this serial schedule is possible under 2PL. It does not violate 2PL rules.**

1. Consider the following schedule that involves transactions T1, T2, and T3.

S: r1(X), r2(Y), r3(Y), w2(Y), w1(X), w3(X), r2(X), w2(X)

Is this schedule possible under 2PL? If yes, then write the corresponding serial schedule.

If no, then provide an equivalent non-serial schedule that is possible under 2PL with minimum swapping of operations.

Lock\_S1(X), Lock\_S2(Y), Lock\_S3(Y), r1(X), r2(Y), r3(Y), unlock3(Y), Lock\_X2(Y), w2(Y), Lock\_X1(X), w1(X), unlock1(X), Lock\_X3(X), w3(X), r2(X), w2(X)

**This schedule is not possible under 2PL since we needed to Lock\_X3(X) after using a unlock3(Y) which violates 2PL (highlighted above).**

**Swap w3(X) and w2(Y):**

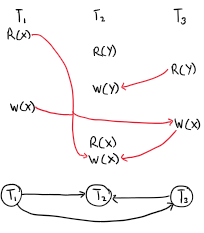
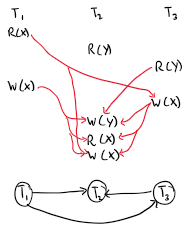
r1(X), r2(Y), r3(Y), **w2(Y)**, w1(X), **w3(X)**, r2(X), w2(X)

**Swap w3(X) and w1(X):**

r1(X), r2(Y), r3(Y), **w3(X)**, **w1(X)**, w2(Y), r2(X), w2(X)

**r1(X), r2(Y), r3(Y), w1(X), w3(X), w2(Y), r2(X), w2(X)** – Possible under 2PL

Lock\_S1(X), Lock\_S2(Y), Lock\_S3(Y), r1(X), r2(Y), r3(Y), Lock\_X1(X), w1(X), unlock1(X), Lock\_X3(X), w3(X), unlock3(X), unlock3(Y), Lock\_X2(Y), Lock\_X2(X), w2(Y), r2(X), w2(X), unlock2(X), unlock2(Y)

1. Consider the following schedules of transactions T1, T2, T3.

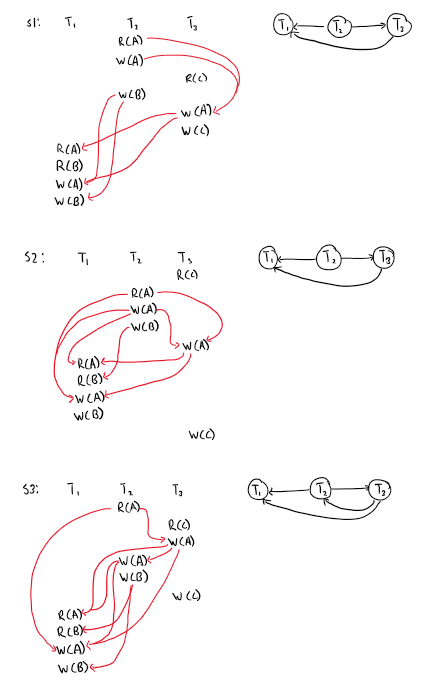
S1: r2(A), w2(A), r3(C), w2(B), w3(A), w3(C), r1(A), r1(B), w1(A), w1(B)

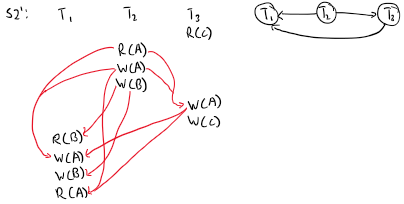
S2: r3(C), r2(A), w2(A), w2(B), w3(A), r1(A), r1(B), w1(A), w1(B), w3(C)

S3: r2(A), r3(C), w3(A), w2(A), w2(B), w3(C), r1(A), r1(B), w1(A), w1(B)

1. Which of the above schedules are conflict equivalent to each other?
2. Which of these schedules are possible in 2PL?
3. For the schedules that are not possible in 2PL, show whether it is possible to swap the operations to get equivalent schedule that is possible under 2PL.

**Answer for Problem #3 \/\/\/\/\/\/\/\/\/\/\/**



1. **S1 and S2 are conflict equivalent and S3 is not conflict serializable.**
2. **Since S3 is not serializable thus it is not possible in 2PL since 2PL can not produce non-serializable schedules.**  
   **Schedule S1 is possible under 2PL**  
   Lock2(A), r2(A), w2(A), Lock3(C), r3(C), Lock2(B), w2(B), unlock2(B), unlock2(A), Lock3(A), Lock3(C), w3(A), w3(C), unlock3(C), unlock3(A), unlock3(C), Lock1(A), Lock1(B), r1(A), r1(B), w1(A), w1(B), unlock1(B), unlock1(A)  
   **Schedule S2 is not possible under 2PL**  
   Lock3(C), r3(C), Lock2(A), Lock2(B), r2(A), w2(A), w2(B), unlock2(B), unlock2(A), Lock3(A), w3(A), unlock3(A), Lock1(A), Lock1(B), r1(A), r1(B), w1(A), w1(B), unlock1(B), unlock1(A), Lock3(C), w3(C), unlock3(C)
3. **S2:  
   r3(C), r2(A), w2(A), w2(B), w3(A), w3(C), r1(B), w1(A), w1(B), r1(A)** – Possible under 2PL  
     
   Lock3(C), Lock2(A), r3(C), r2(A), Lock2(B), w2(A), w2(B), unlock2(B), unlock2(A), Lock3(A), w3(A), w3(C), unlock3(A), unlock(C), Lock1(B), Lock1(A), r1(B), w1(A), w1(B), r1(A), unlock1(A), unlock1(B)  
     
     
   **S3:  
   Since S3 is not a conflict serializable schedule therefore S3 can not be a product of 2PL therefore there cannot be a way to swap operations to get a schedule that is possible under 2PL.**
4. S: r1(Y), w1(Y), r2(Y), w2(Y), r1(X), w1(X), r2(X), w2(X). Is this schedule possible in 2PL? Explain your answer. Produce a schedule with the same set of operations that will cause a deadlock.

**Lock1(Y), r1(Y), w1(Y), unlock1(Y), Lock2(Y), r2(Y), w2(Y), Lock1(X), r1(X), w1(X), unlock1(X), Lock2(X), r2(X), w2(X), unlock2(X), unlock2(Y)**

**This schedule is not possible in 2PL since there was a Lock1 after an unlock1 which violates 2PL (highlighted above).**

**r1(Y), w1(Y), r2(X), w2(Y), r1(X), w1(X), r2(Y), w2(X) – will cause a deadlock**

1. T1= read(A), A:=A+5, write(A), read(B), B:=B-10

T2= read(A), A:=A-10, write(A), read(B), B:=B+5

T3= read(A), A:=A+3, write(A), read(B), B:=B-2

A schedule is equivalent (not conflict equivalent, just equivalent) to a serial schedule S if it leaves the database in the same state as S (items have the same values)

Is there a schedule S for T1, T2 and T3 which is not equivalent to some serial schedule? If so, show such schedule. If not, explain why.

**No, for any serial schedule that is generated by T1, T2, and T3 all the schedules are equivalent. Each transaction will completely finish before the next transaction starts therefore in any order that T1, T2, and T3 is in will be equivalent.**

**Data item [B]**

**Since B is only read and never written, B will never change in the database thus leaving the database the same.**

**Data item [A]**

**Having to read in A then doing an operation on A then writing it to the database in different orders does not have any affect on the result of A. For example, if the transactions are ordered T1, T2, T3 (assuming A is 0) the resulting A will be (((0+5)-10)+3) = -2. The result will be the same if the transaction order was T2, T1, T3 (again assuming A is 0) (((0-10)+5)+3) = -2.**

1. Suppose for the new bar New Tavern has a remote order system which allows customers order beers by themselves. The inventory information of New Tavern is shown in the table:

|  |  |  |
| --- | --- | --- |
| Bar | Beer | Inventory |
| Tavern | Budweiser | 13 |
| Tavern | Heineken | 22 |
| Tavern | Pabst Blue Ribbon | 3 |
| Tavern | Corona | 9 |

There are three customers A, B and C ordering beers remotely, they start their transactions in the given order:

1. Customer A orders two Budweisers:

1. Customer B orders one Budweiser and one Pabst Blue Ribbon:

1. Customer C orders two Coronas:

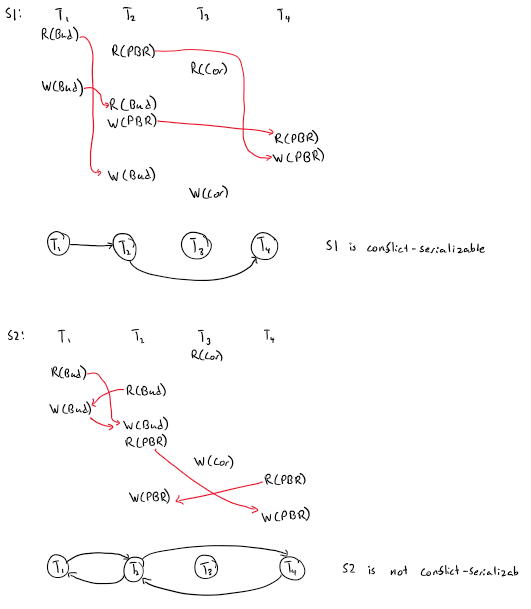
1. Customer D orders three Pabst Blue Ribbons:

Let

1. r1(Bud)r2(PBR)r3(Cor)w1(Bud)r2(Bud)w2(PBR)r4(PBR)w4(PBR)w2(Bud)w3(Cor)
2. r3(Cor)r1(Bud)r2(Bud)w1(Bud)w2(Bud)r2(PBR)w3(Cor)r4(PBR)w2(PBR)w4(PBR)

Q1: For schedules and , which one is conflict-serializable? Which one is not conflict-serializable? Please justify your answer with Precedence Graph.

**Answer to Q1 \/\/\/\/\/\/\/\/\/\/\/\/**



Q2: For the schedule that is not conflict-serializable, show it will not be granted under 2PL protocol.

**Lock\_S3(Cor), Lock\_S1(Bud), Lock\_S2(Bud), r3(Cor), r1(Bud), r2(Bud), Lock\_X1(Bud), w1(Bud), unlock1(Bud), Lock\_X2(Bud), Lock\_S2(PBR), w2(Bud), r2(PBR), Lock\_X3(Cor), w3(Cor), unlock3(Cor), unlock2(PBR), Lock\_S4(PBR), r4(PBR), unlock4(PBR), Lock\_X2(PBR), w2(PBR), unlock2(PBR), Lock\_X4(PBR), w4(PBR)**

**There was a Lock\_X2 after an unlock2 which violates 2PL protocol (Highlighted above).**

Q3: Schedule that is not conflict-serializable, what troubles would it cause for New Tavern?

**The number of beers in inventory will be wrong since there are blind writes to the database from different customers. For example, in the transaction with Customer A and B both transactions read in the number of Budweiser’s which is 13. Transaction 1 writes to the database (13-2) = 11 since Customer A ordered 2 Budweiser’s. But then Transaction 2 does a blind write to the database without reading in the new value of Budweiser’s and overwrites the value to (13-1) = 11 since Customer B ordered only 1 Budweiser. The correct amount of Budweiser’s should be (13-3) = 10 instead of 11. The same thing occurs with Customer B and Customer D where Transaction 4 does a blind write for the number of Pabst Blue Ribbons resulting in the number of beers going down to 0 instead of -1. (Since Customer B ordered 1 Pabst Blue Ribbon and Customer D ordered 3 which totals up to 4 orders of Pabst Blue Ribbons)**