COMPUTER SCIENCE

Database Management System

Transaction & Concurrency Control

Serializable Schedule (Conflict & View) Part-02

Lecture_4

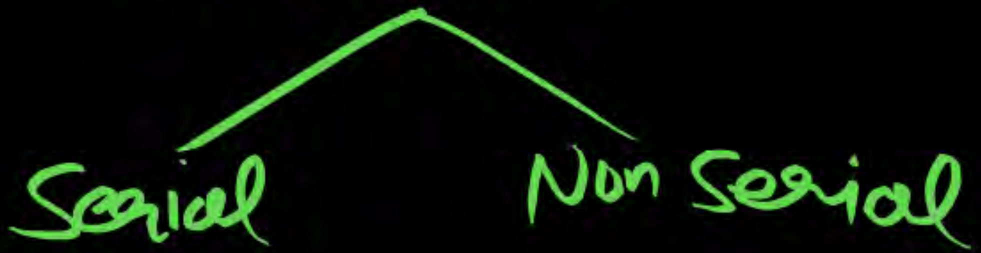
Vijay Agarwal sir

TOPICS TO BE COVERED

01 Conflict Serializable

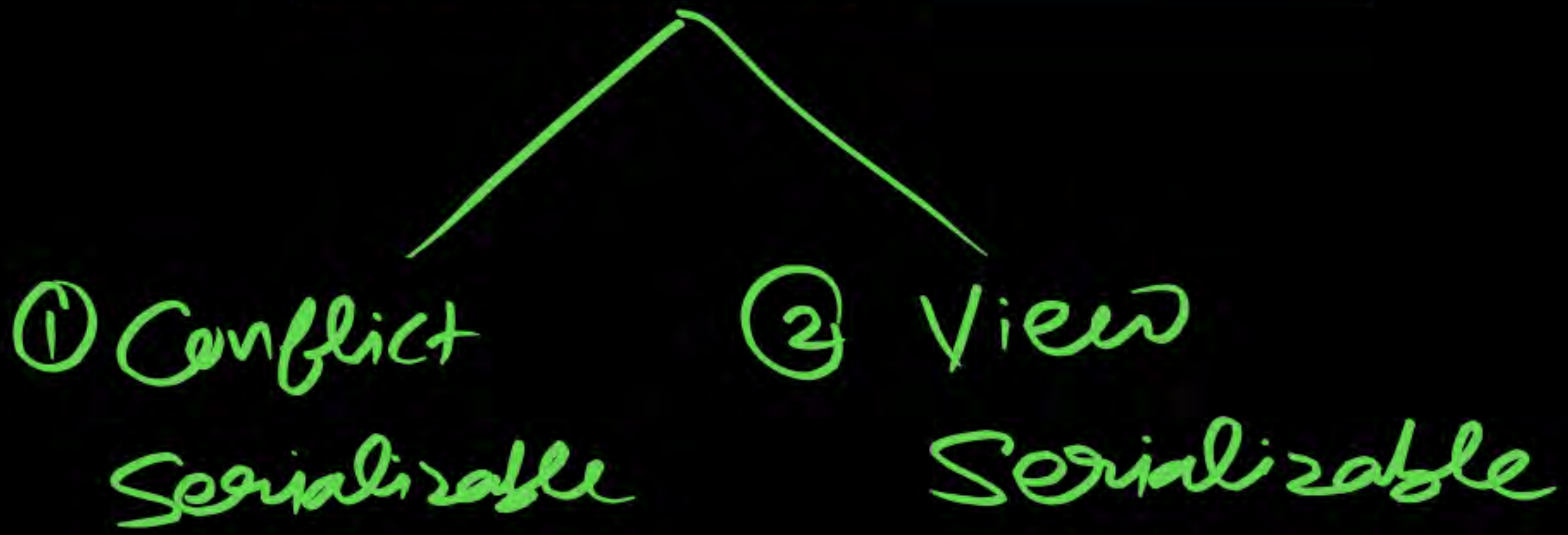
02 View Serializable

- Transaction Concept
- ACID Properties
- Transaction States
- Schedule


Serial

Non Serial

serializable Schedule


① Conflict Serializable

② View Serializable

conflict serializable

Testing for Conflict serializable

→ Precedence Graph Method.

Precedence Graph method

G : (V, E) Directed Graph

(vertex) V: Set of transaction

(Edge) E: Directed Edge. $T_i \rightarrow T_j$

CNC

Cycle Not Conflict

$T_i \rightarrow T_j$

R(A) – W(A)

W(A) – R(A)

W(A) – W(A)

then create Arc (Edge)

Conflict Serializable

A schedule is said to be conflict serializable if it is conflict equivalent to a serial schedule.

Same conflicting operation order in C_1 & S_1

∴ Its $\{C_1\}$ conflict is conflict serializable.

T_1	T_2
read(A)	
write(A)	
	read(A)
	write(A)
read(B)	
write(B)	
	read(B)
	write(B)

C_L

T_1	T_2
read(A)	
write(A)	
read(B)	
write(B)	
	read(A)
	write(A)
	read(B)
	write(B)

S_L

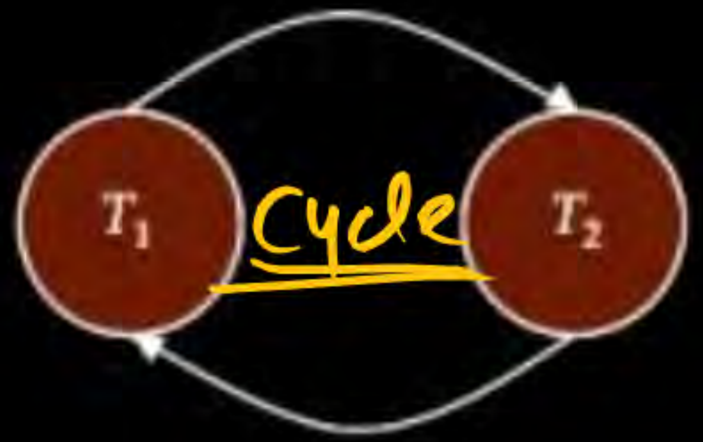
Conflicting Instructions

- Instructions l_i, and l_j of transactions T_i and T_j respectively, conflict if and only if there exists some item Q accessed by both l_i, and l_j, and at least one of these instructions wrote Q.
  1. l_i, = read(Q), l_j = read(Q). l_i and l_j don't conflict.
  2. l_i, = read(Q) l_j = write(Q). They conflict.
  3. l_i, = write(Q) l_j = read(Q). They conflict
  4. l_i= write(Q) l_j = write(Q). They conflict
- Intuitively, a conflict between l_i and l_j forces a (logical) temporal order between them.
  - If l_i, and l_j are consecutive in a schedule and they do not conflict, their results would remain the same even if they had been interchanged in the schedule.

Testing for Serializability

- Testing for conflict serializability.
  - Consider some schedule of a set of transactions T_1, T_2, ...T_n
  - Precedence graph — a direct graph where the vertices are the transactions (names).
  - We draw an arc from T_i to T_j if the two transaction conflict, and T_i accessed the data item on which the conflict arose earlier.
  - We may label the arc by the item that was accessed.

Example:



A schedule is conflict serializable if and only if its precedence graph is acyclic.

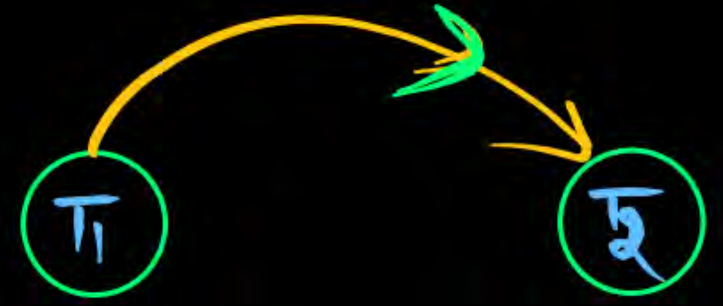
If Graph Contain Any One Cycle (if cycle exists between Any Two Transaction) then Schedule is Not Conflict Serializable.

NOTE: CNC [Cycle not conflict serializable]

Q.1

S: $R_1(A)\ W_1(A)\ R_2(A)\ W_2(A)\ R_1(B)\ W_1(B)\ R_2(B)\ W_2(B)$

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
R(B)	
W(B)	
	R(B)
	W(B)



Conflict Serializable

<T_1 T_2>

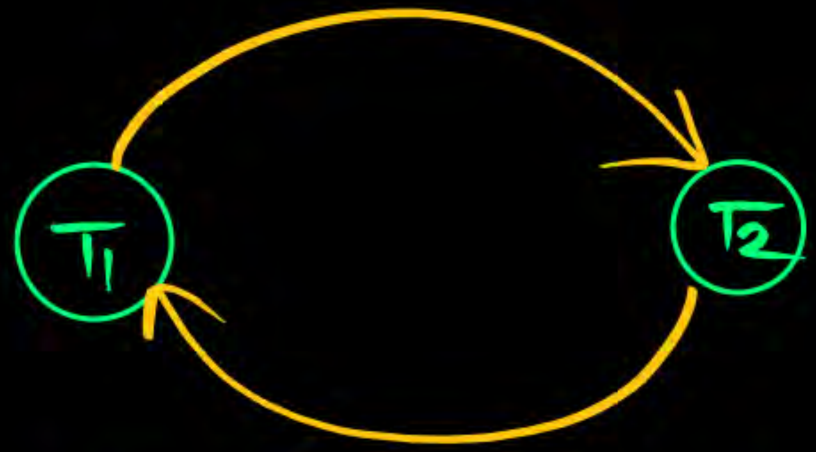
T_1 followed by T_2.

Q.2

$R_1(A)\ R_2(A)\ W_2(A)\ W_1(A)\ R_1(B)\ W_1(B)\ R_2(B)\ W_2(B)$

T_1	T_2
R(A)	
	R(A)
	W(A)
W(A)	
R(B)	
W(B)	
	R(B)
	W(B)

CNC



Cycle Not Conflict

ⓐ cc with enjoying

ⓑ cc

ⓒ c

ⓓ Doubt

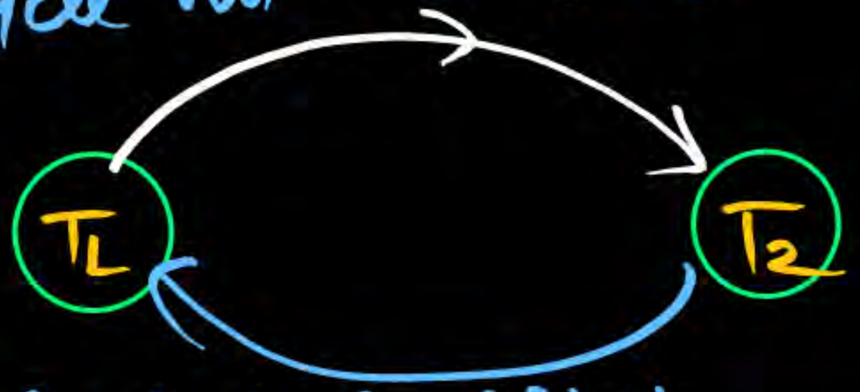
Q.3

$R_1(A)\ W_1(A)\ R_2(B)\ W_2(B)\ R_1(B)\ W_1(B)\ R_2(A)\ W_2(A)$

T1	T2
R(A)	
W(A)	
	R(B)
	W(B)
R(B)	
W(B)	
	R(A)
	W(A)

CNC

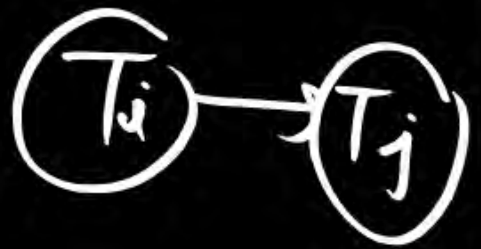
Cycle Not Conflict



Cycle Not Conflict

Each Data Item (x)

Ti → Tj (i ≠ j)



R(X) → W(X)

W(X) → R(X)

W(X) → W(X)

Serializability Order

Important Point 1:

1. If S_1, S_2 Schedule are conflict equal then precedence graph of S_1 and S_2 must be same.
2. If S_1 and S_2 have same precedence graph then S_1 and S_2 may or may not conflict equal.

Q.4

Consider the following schedules involving two transactions. Which one of the following statements is TRUE?

S_1: $r_1(X); r_1(Y); r_2(X); r_2(Y); w_2(Y); w_1(X)$

S_2: $r_1(X); r_2(X); r_2(Y); W_2(Y); r_1(Y); w_1(X)$

[2007: 2 Marks]

A Both S_1 and S_2 are conflict serializable

Ans [c]

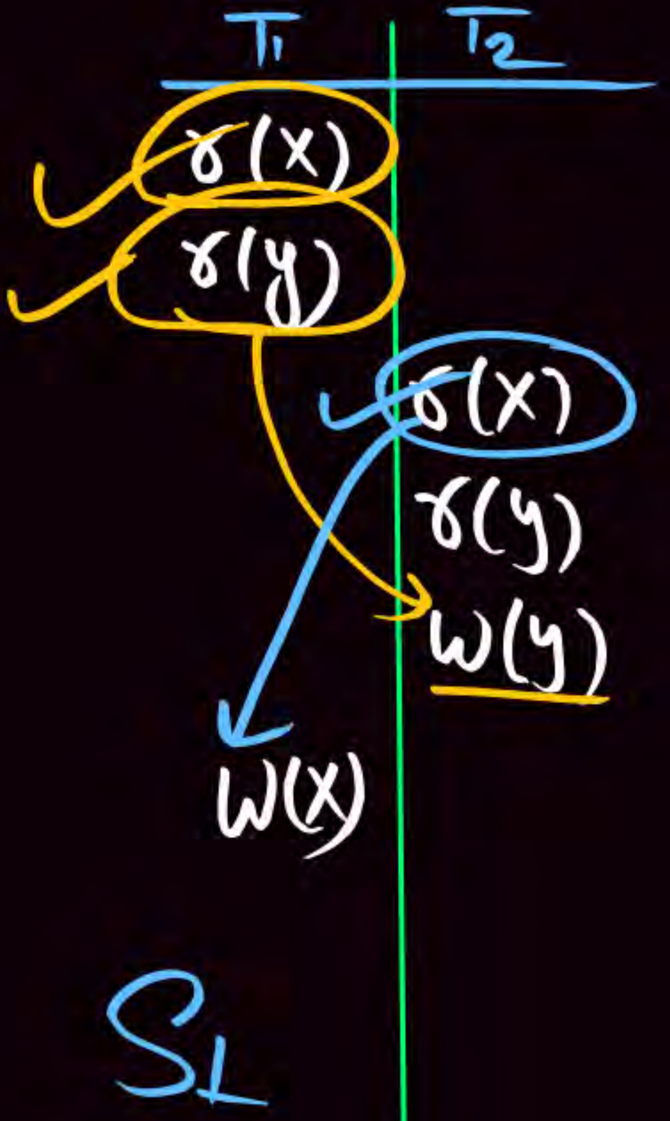
B S_1 is conflict serializable and S_2 is not conflict serializable

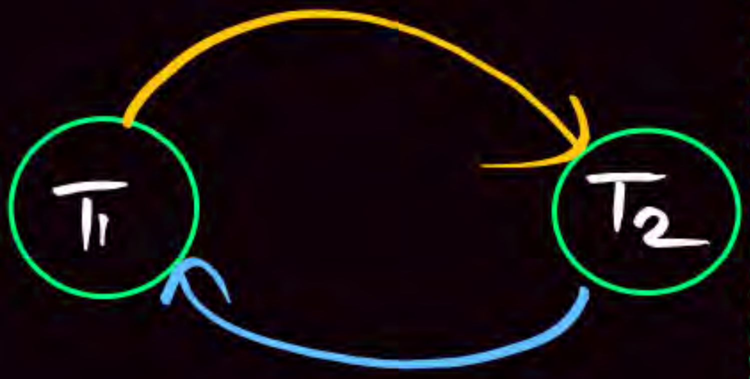
[✓ C] S_1 is not conflict serializable and S_2 is conflict serializable

D Both S_1 and S_2 are not conflict serializable

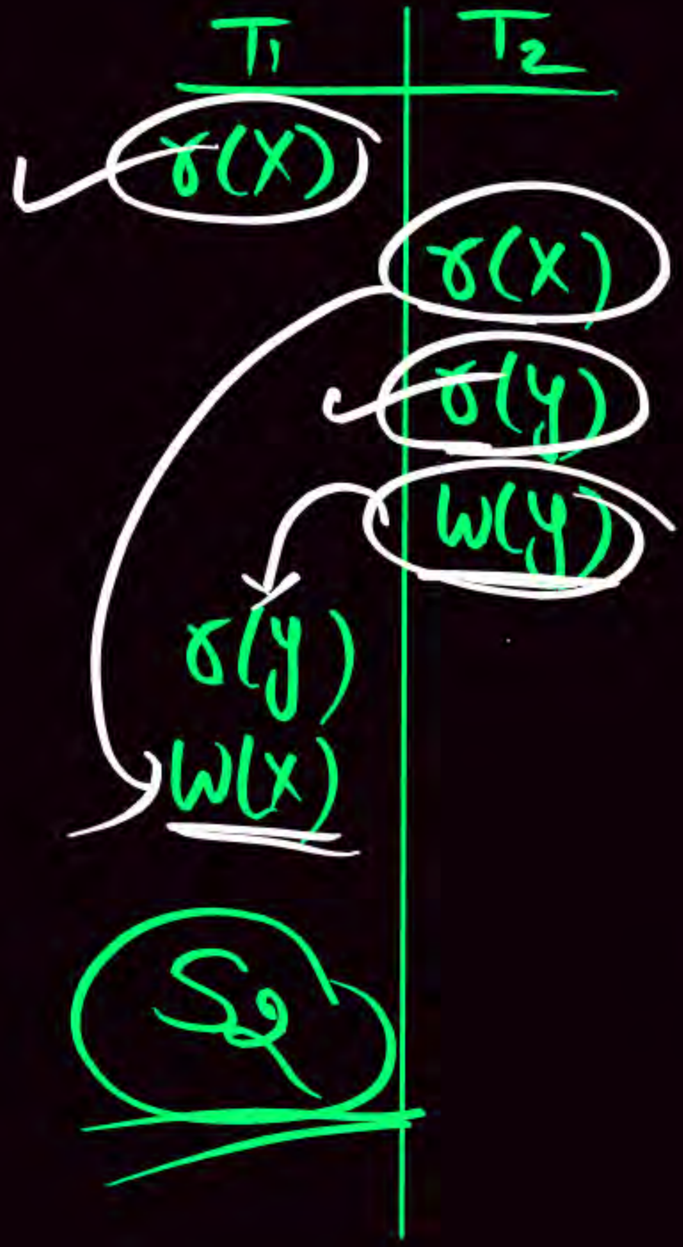
$S_1: r_1(X)\; r_1(y)\; r_2(X)\; r_2(y)\; w_2(y)\; w_1(X)$

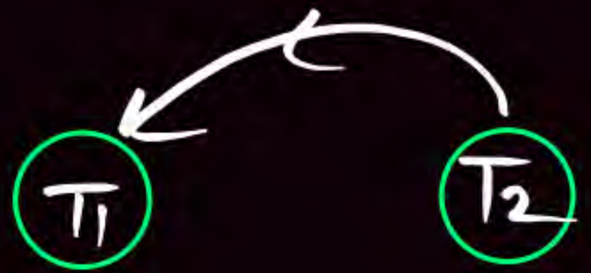
$S_2: r_1(X)\; r_2(X)\; r_2(y)\; w_2(y)\; r_1(y)\; w_1(X)$





S1: Cycle Not Conflict





Conflict serializable

< T2 T1 >

T2 followed by T1.

Q.5.

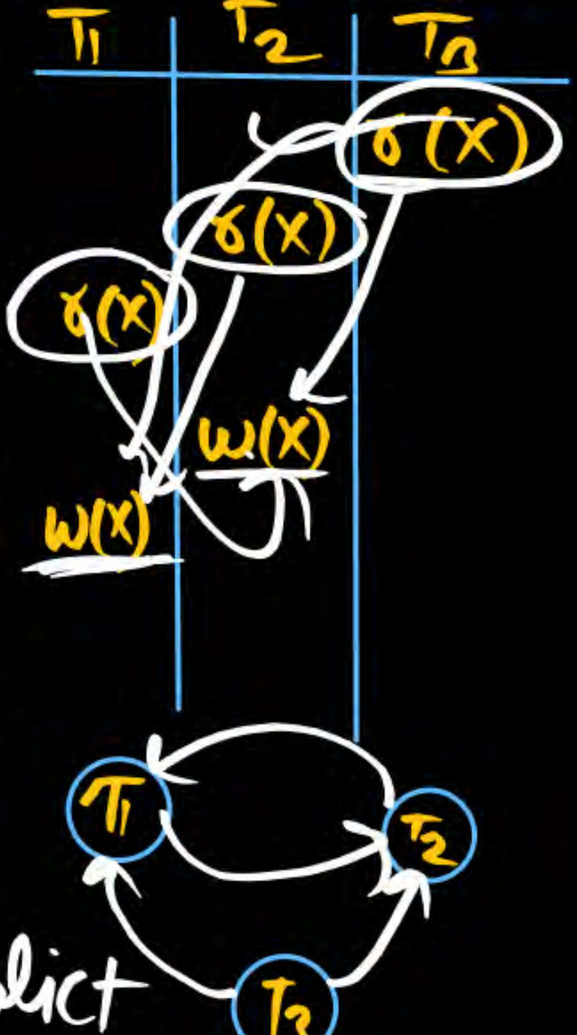
Consider the following four schedules due to three transactions (indicated by the subscript) using read and write on a data item x, denoted by r(x) and w(x) respectively. Which one of them is conflict serializable?

[2014(Set-1): 2 Marks]

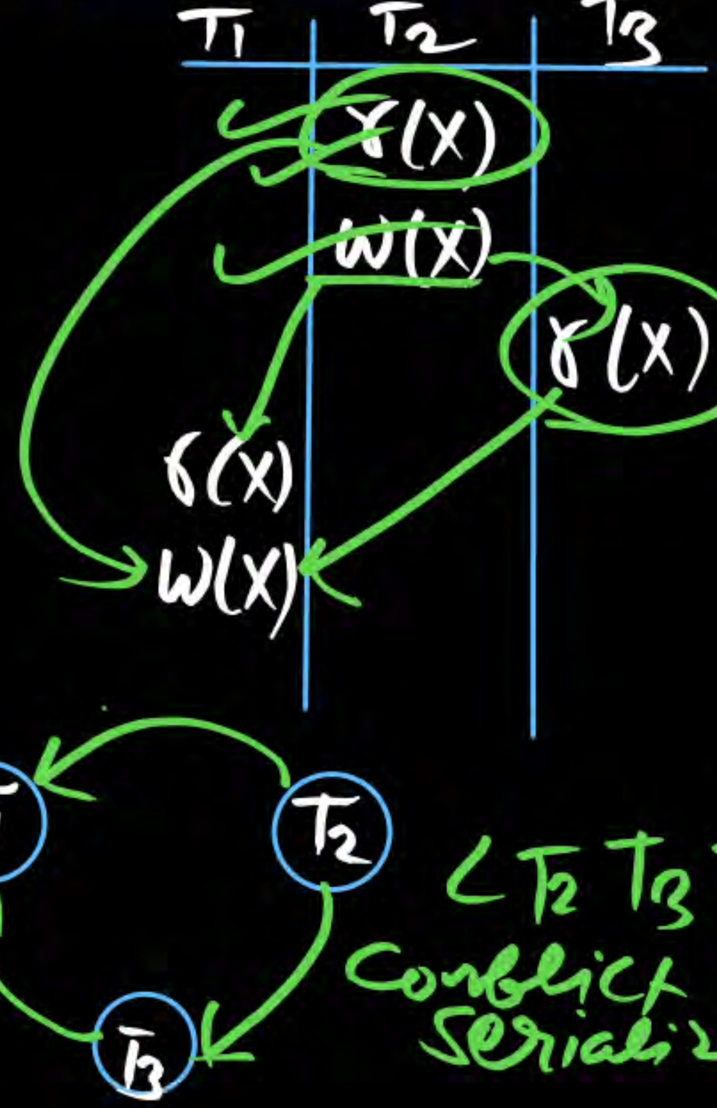
- ~~(A)~~ $r_1(x); r_2(x); w_1(x); r_3(x); w_2(x)$
- ~~(B)~~ $r_2(x); r_1(x); w_2(x); r_3(x); w_1(x)$
- ~~(C)~~ $r_3(x); r_2(x); r_1(x); w_2(x); w_1(x)$
- [✓ (D)] $r_2(x); w_2(x); r_3(x); r_1(x); w_1(x)$

$\boxed{\text{Ans (D)}}$

Ⓒ

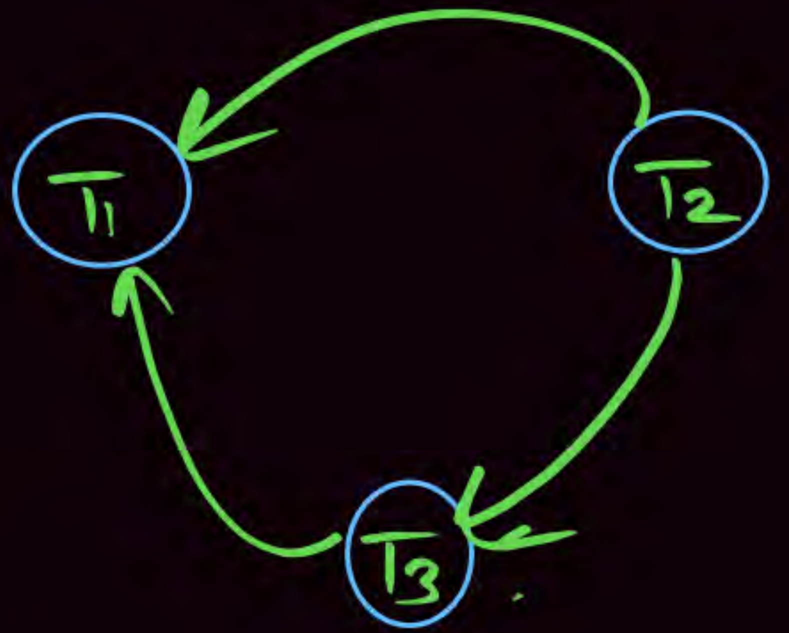


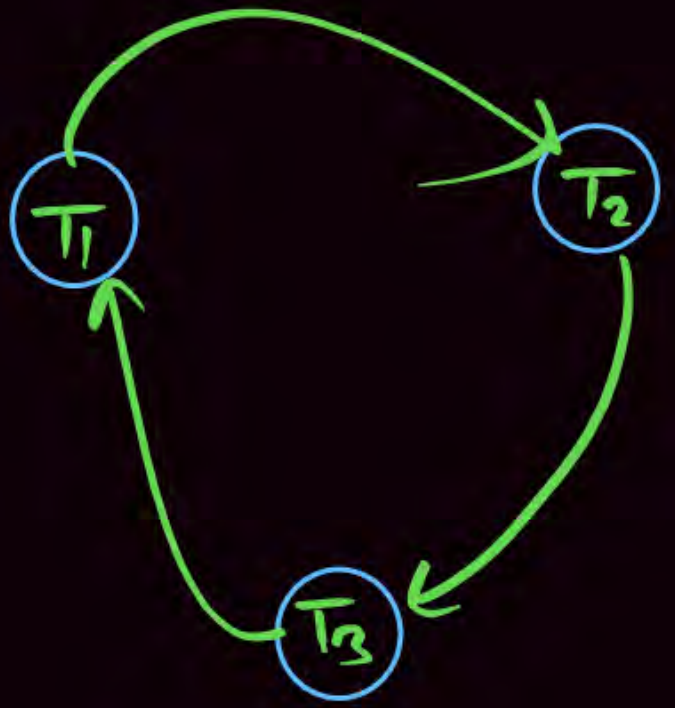
Cycle
Not Conflict

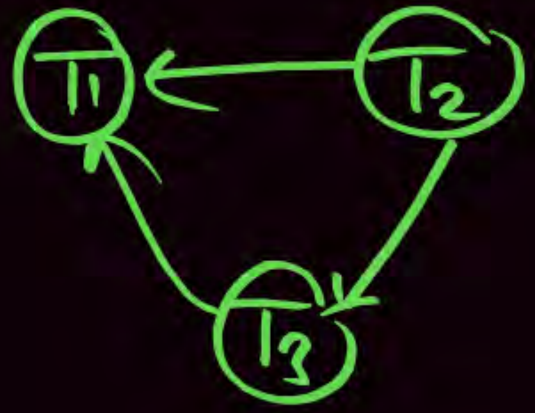


<T2 T3 T1>
Conflict
Serializable

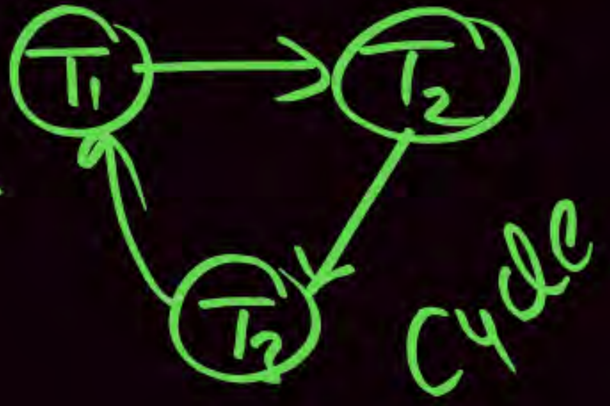
d



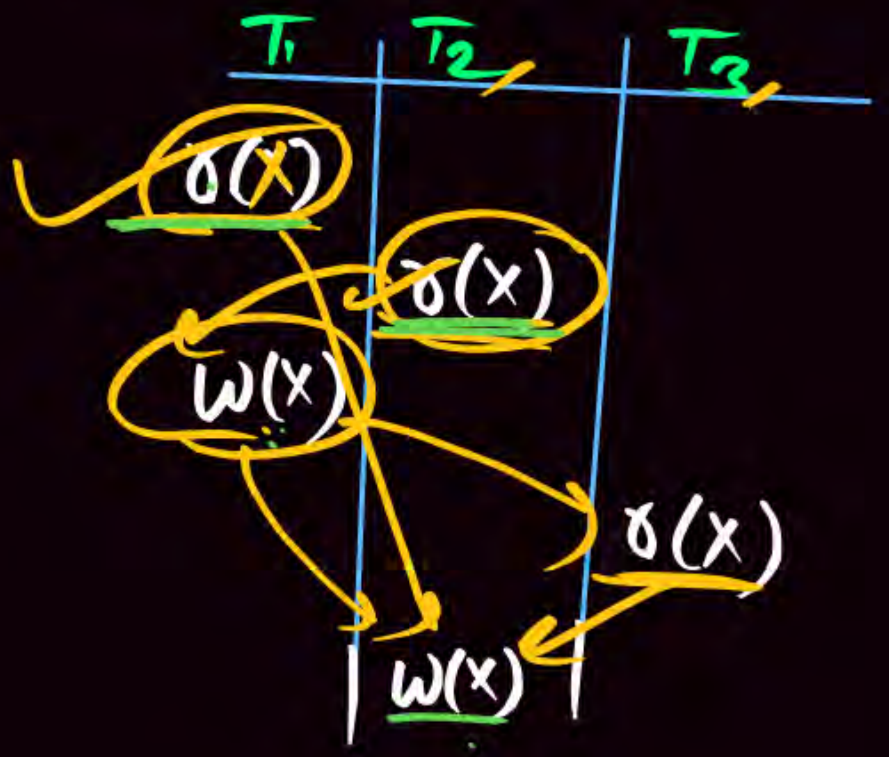




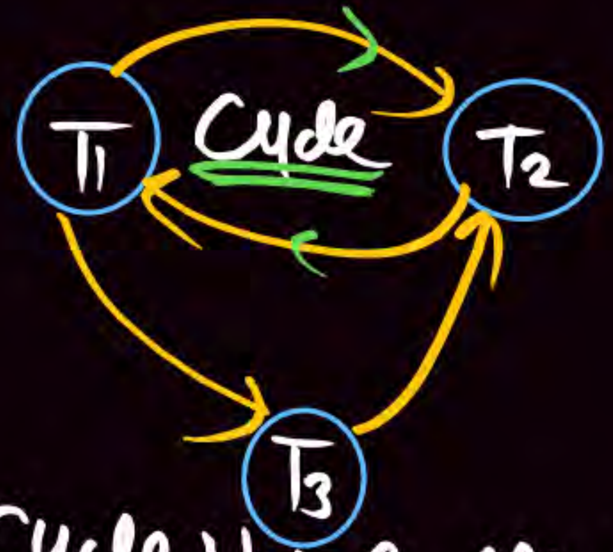
Topological Sorting



ⓐ $r_1(x)$ $r_2(x)$ $w_1(x)$ $r_3(x)$ $w_2(x)$

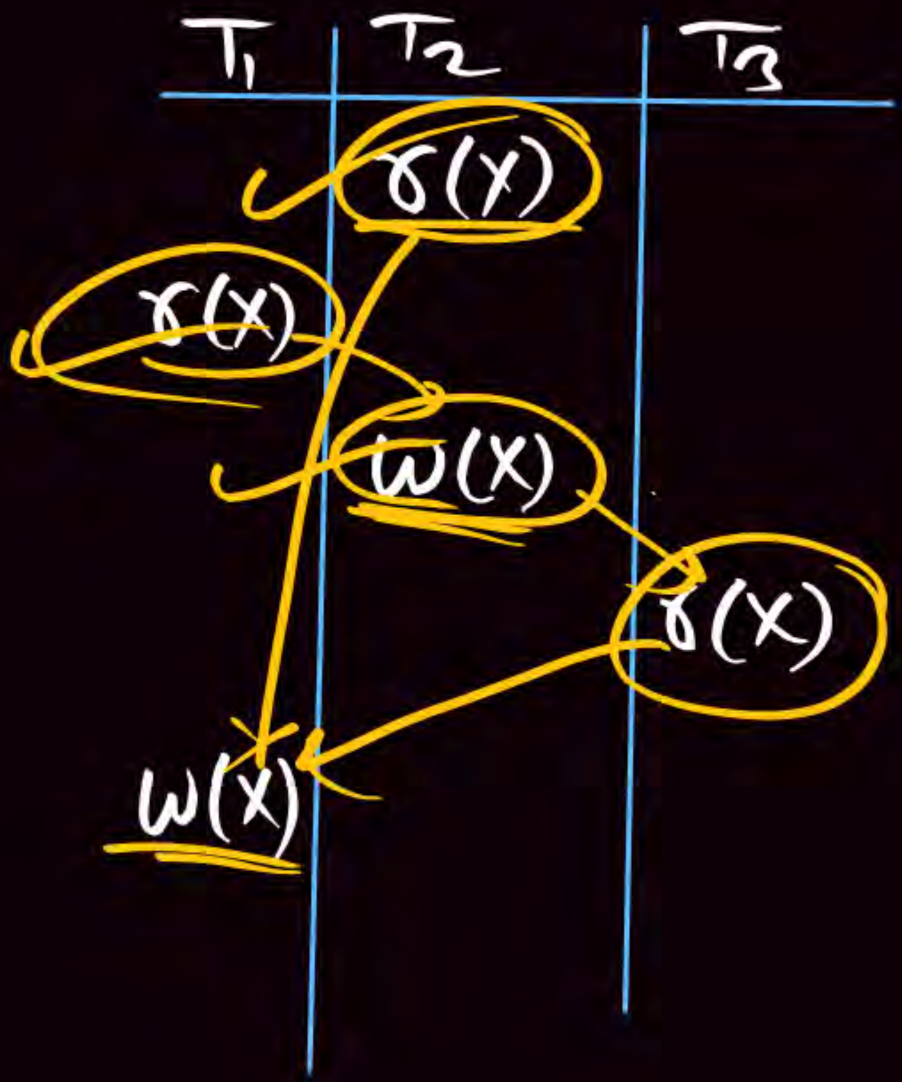


T_1	T_2	T_3
r(x)		
	r(x)	
w(x)		
		r(x)
	w(x)	

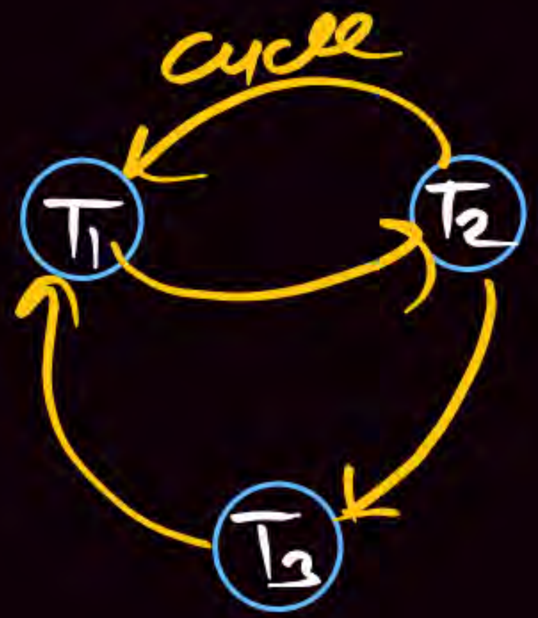


Cycle Not Conflict

ⓑ $r_2(x)$ $r_1(x)$ $w_2(x)$ $r_3(x)$ $w_1(x)$



T_1	T_2	T_3
	r(x)	
r(x)		
	w(x)	
		r(x)
w(x)		



Cycle Not Conflict

Q. 6.

Let $r_i(z)$ and $w_i(z)$ denote read and write operations respectively on a data item by a transaction T_i. Consider the following two schedules.

S_1: $r_1(x)\ r_1(y)\ r_2(x)\ r_2(y)\ w_2(y)\ w_1(x)$

S_2: $r_1(x)\ r_2(x)\ r_2(y)\ w_2(y)\ r_1(y)\ w_1(x)$

Which one of the following options is correct? [MCQ: 2021: 2M]

A. S_1 is conflict serializable, and S_2 is not conflict serializable.

[✓ B.] S_1 is not conflict serializable, and S_2 is conflict serializable.

C. Both S_1 and S_2 are conflict serializable.

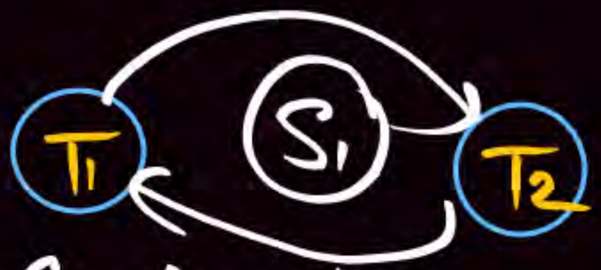
D. Neither S_1 nor S_2 is conflict serializable.

$\boxed{\text{Ans (B)}}$

$S_1: r_1(X)\ r_1(y)\ r_2(X)\ r_2(y)\ w_2(y)\ w_1(X)$

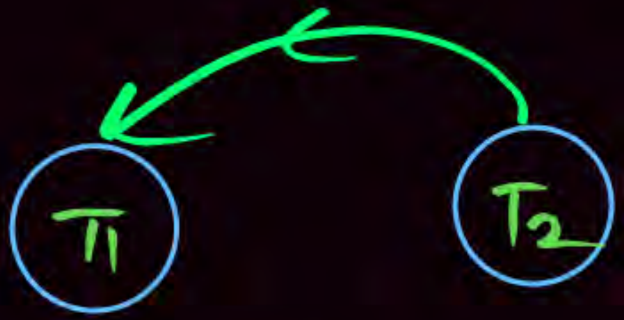
$S_2: r_1(X)\ r_2(X)\ r_2(y)\ w_2(y)\ r_1(y)\ w_1(X)$

T1	T2
r(X)	
r(y)	
	r(X)
	r(y)
	W(y)
W(X)	



Cycle Not Conflict.

T1	T2
r(X)	
	r(X)
	r(y)
	W(y)
r(y)	
W(X)	

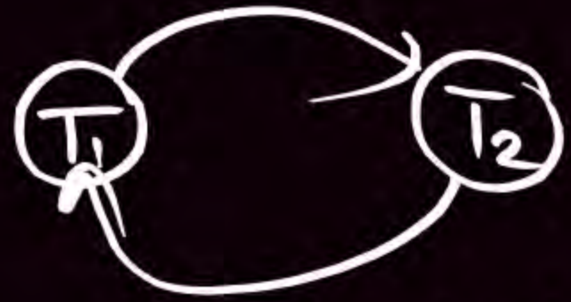


< T2 , T1 >

Conflict Serializable

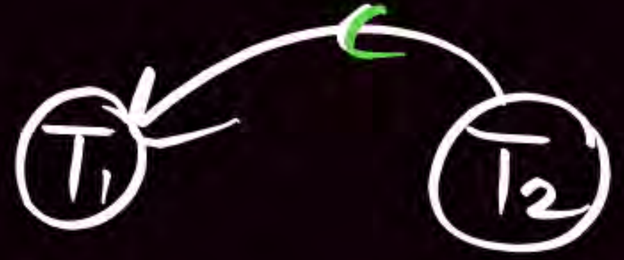
Alternate (IInd Method)

S_1 : $r_1(X)$ $r_1(y)$ $r_2(X)$ $r_2(y)$ $w_2(y)$ $w_1(X)$



Cycle Not Conflict

S_2 : $r_1(X)$ $r_2(X)$ $r_2(y)$ $w_2(y)$ $r_1(y)$ $w_1(X)$



< T2 T1 >

Conflict serializable.

Note If Precedence Graph Contain Any cycle then Schedule is Not Conflict Serializable [CNC].

Note If Precedence graph does not Contain Any cycle then Schedule is Conflict Serializable. (then Serializability order check).

Note If schedule is Conflict Serializable then its means its Conflict Equivalent to Any Serial Schedule.

Note This Serializability order is Determined by Topological sorting

Note

Serializablity order tells you this concurrent execution is equivalent to which serial schedule of the Given schedule.

Topological Sorting. [finding Serializability order] ≡ (equivalent serial schedule)

Topological Sorting. : Starts from the vertex which having Indegree is '0' (No Incomming edge).

& then Delete the Connected edge.

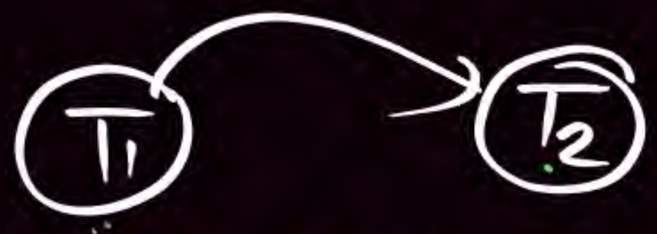
& Repeat the Steps & so on.

finding

Topological Sorting. [Serializablity Order] ≡ (equivalent serial schedule)

→ Always for Acyclic graph

eg1



Serializablity order : $\langle T_1 , T_2 \rangle$

T_1 followed by T_2.

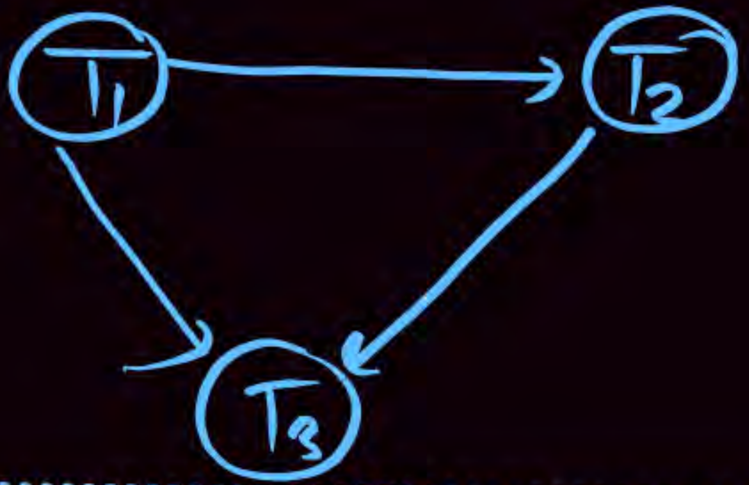
Conflict serializable

$\langle T_1 , T_2 \rangle$

T_1 followed by T_2 means this Concurrent Schedule Result is equal to Serial Schedule T_1 followed by T_2 $\langle T_1 , T_2 \rangle$.

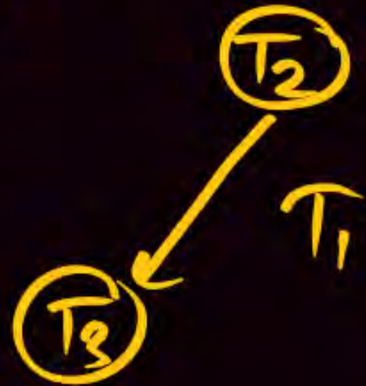
Serializablity order / Topological sorting

eg2



<T1 T2 T3>

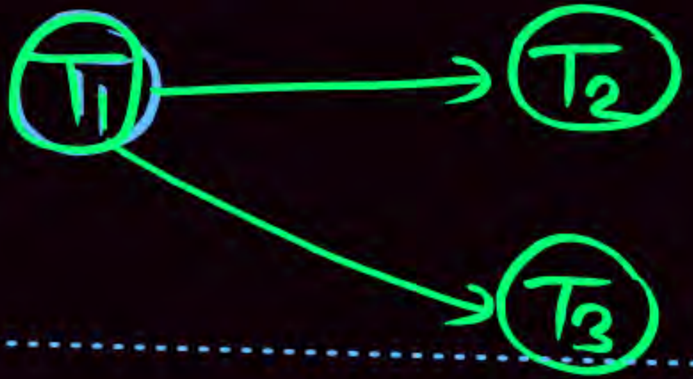
OR



T1 Indegree
'0'

<T1, T2, T3>

eg3

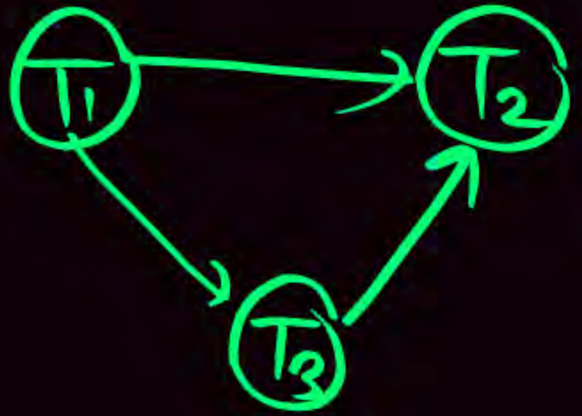


<T1 T2 T3>

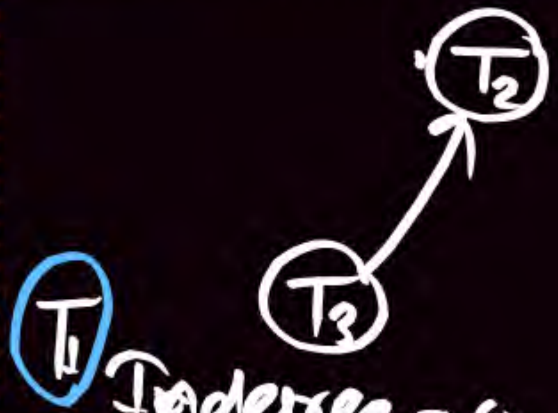
OR

<T1 T3 T2>

eg4



<T1 T3 T2>



Indegree = 0

<T1, T3, T2>

Q.

Let $R_i(z)$ and $W_i(z)$ denote read and write operations on a data element z by a transaction T_i, respectively. Consider the schedule S with four transactions.

S: $R_4(x), R_2(x), R_3(x), R_1(y), W_1(y), W_2(x), W_3(y), R_4(y)$

Which one of the following serial schedules is conflict equivalent to S?

[2022: 2 Marks]

[✓ A] $T_1 \rightarrow T_3 \rightarrow T_4 \rightarrow T_2$

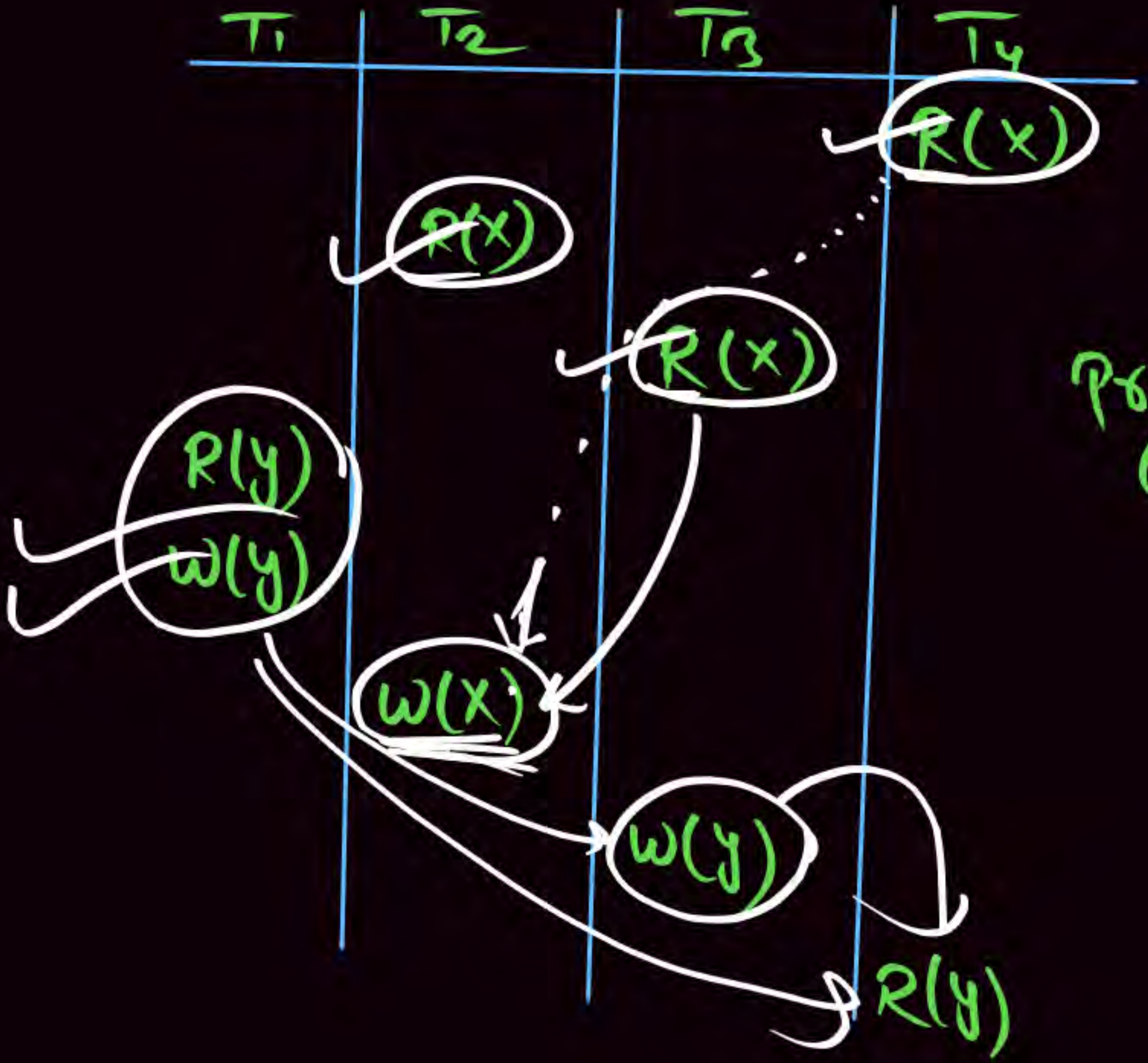
B $T_1 \rightarrow T_4 \rightarrow T_3 \rightarrow T_2$

C $T_4 \rightarrow T_1 \rightarrow T_3 \rightarrow T_2$

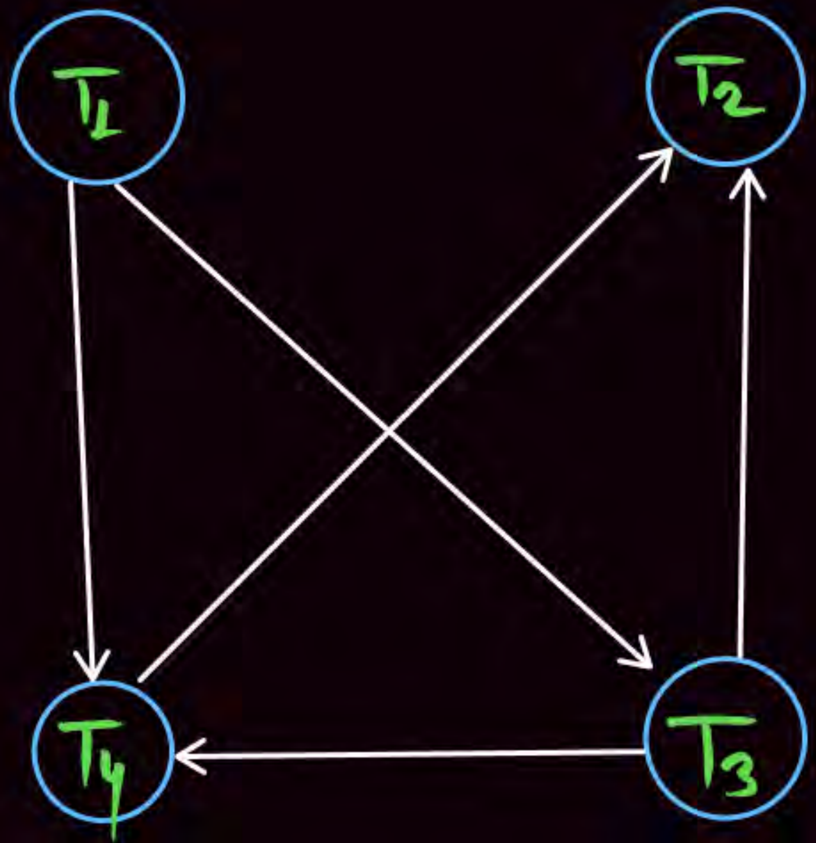
D $T_3 \rightarrow T_1 \rightarrow T_4 \rightarrow T_2$

Ans (a).

$R_4(X)\ \ R_2(X)\ \ R_3(X)\ \ R_1(y)\ \ W_1(y)\ \ W_2(X)\ \ W_3(y)\ \ R_4(y)$



Precedence Graph



$\langle T_1, T_3, T_4, T_2 \rangle$.

$R_4(X)$ $R_2(X)$ $R_3(X)$ $R_1(y)$ $W_1(y)$ $W_2(X)$ $W_3(y)$ $R_4(y)$

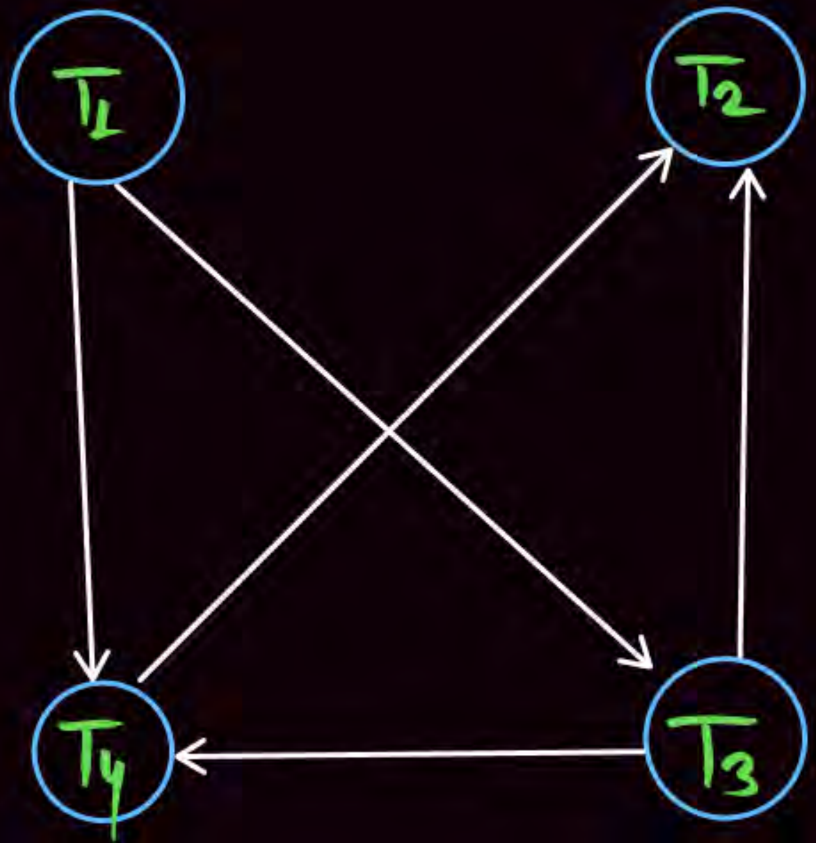
Part I

Topological Sorting

T_1 Indegree = 0

$< T_1$

Next Slide



$R_4(x)$ $R_2(x)$ $R_3(x)$ $R_1(y)$ $W_1(y)$ $W_2(x)$ $W_3(y)$ $R_4(y)$

Part II

Topological Sorting

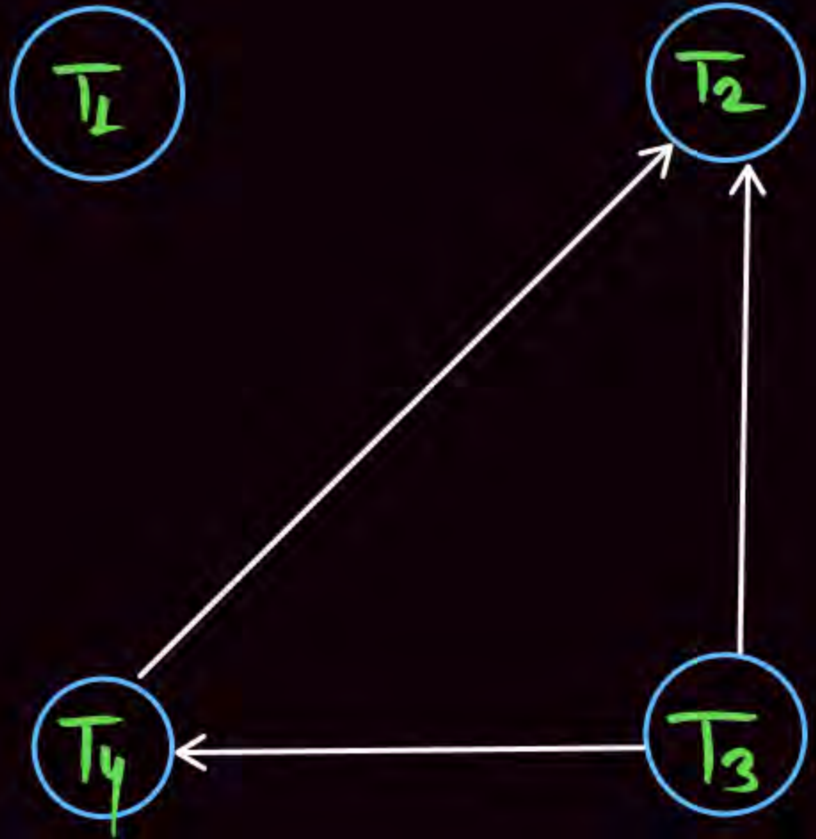
T_1 Indegree = 0

$< T_1$

Now T_3 Indegree = 0

$< T_1, T_3,$

Next slide.



$R_4(X)\ R_2(X)\ R_3(X)\ R_1(y)\ W_1(y)\ W_2(X)\ W_3(y)\ R_4(y)$

Part III

Topological Sorting

T_1 Indegree = 0

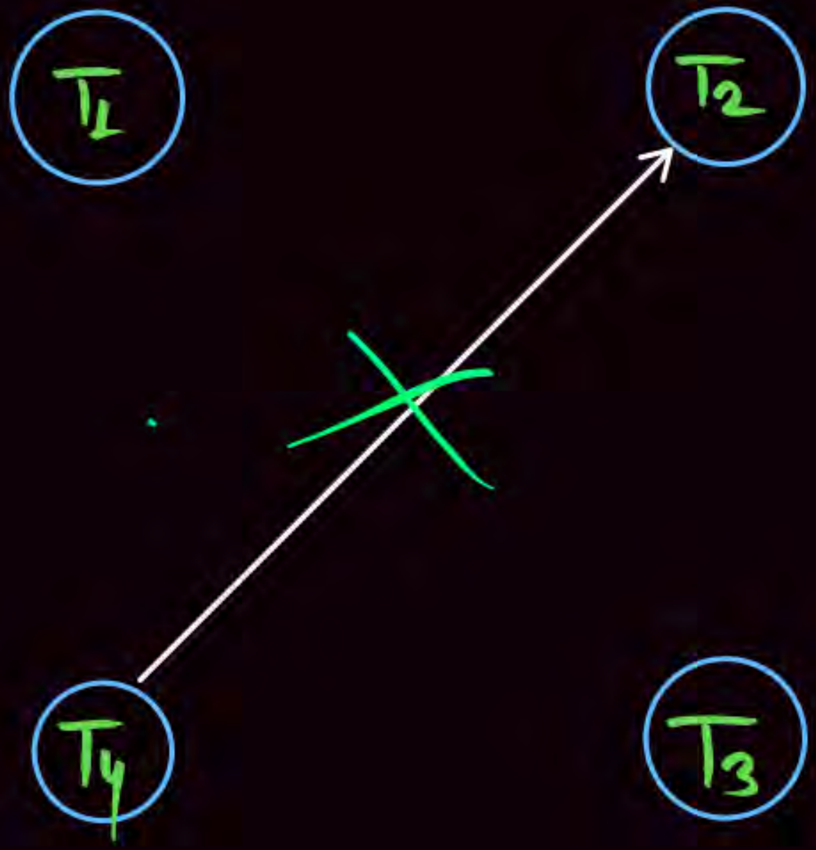
< T_1

Now T_3 Indegree = 0

< T_1, T_3,

Now then T_4 Indegree = 0

< T_1, T_3, T_4, T_2 >.



IInd (Alternate Method)

$R_4(X) \quad R_2(X) \quad R_3(X) \quad R_1(y) \quad W_1(y) \quad W_2(X) \quad W_3(y) \quad R_4(y)$

Conflict operation

for Data Item X : $R_4(X) \rightarrow W_2(X) \;\therefore\; T_4 \rightarrow T_2$

$R_3(X) \rightarrow W_2(X) : \therefore\; T_3 \rightarrow T_2$.

for Data Item y :

$R_1(y) - W_3(y) : \therefore\; T_1 \rightarrow T_3$

$W_1(y) - W_3(y) : \therefore\; T_1 \rightarrow T_3$

$W_1(y) - R_4(y) : \therefore\; T_1 \rightarrow T_4$

$W_3(y) - R_4(y) : \therefore\; T_3 \rightarrow T_4$.

$\boxed{< T_1, T_3, T_4, T_2 >}$

Q.

Consider the following transaction involving two bank accounts x and y.

read(x); x: = x – 50; write (x); read (y); y: = y + 50; write (y)

The constraint that the sum of the accounts x and y should remain constant is that of [2015(Set-2): 1 Marks]

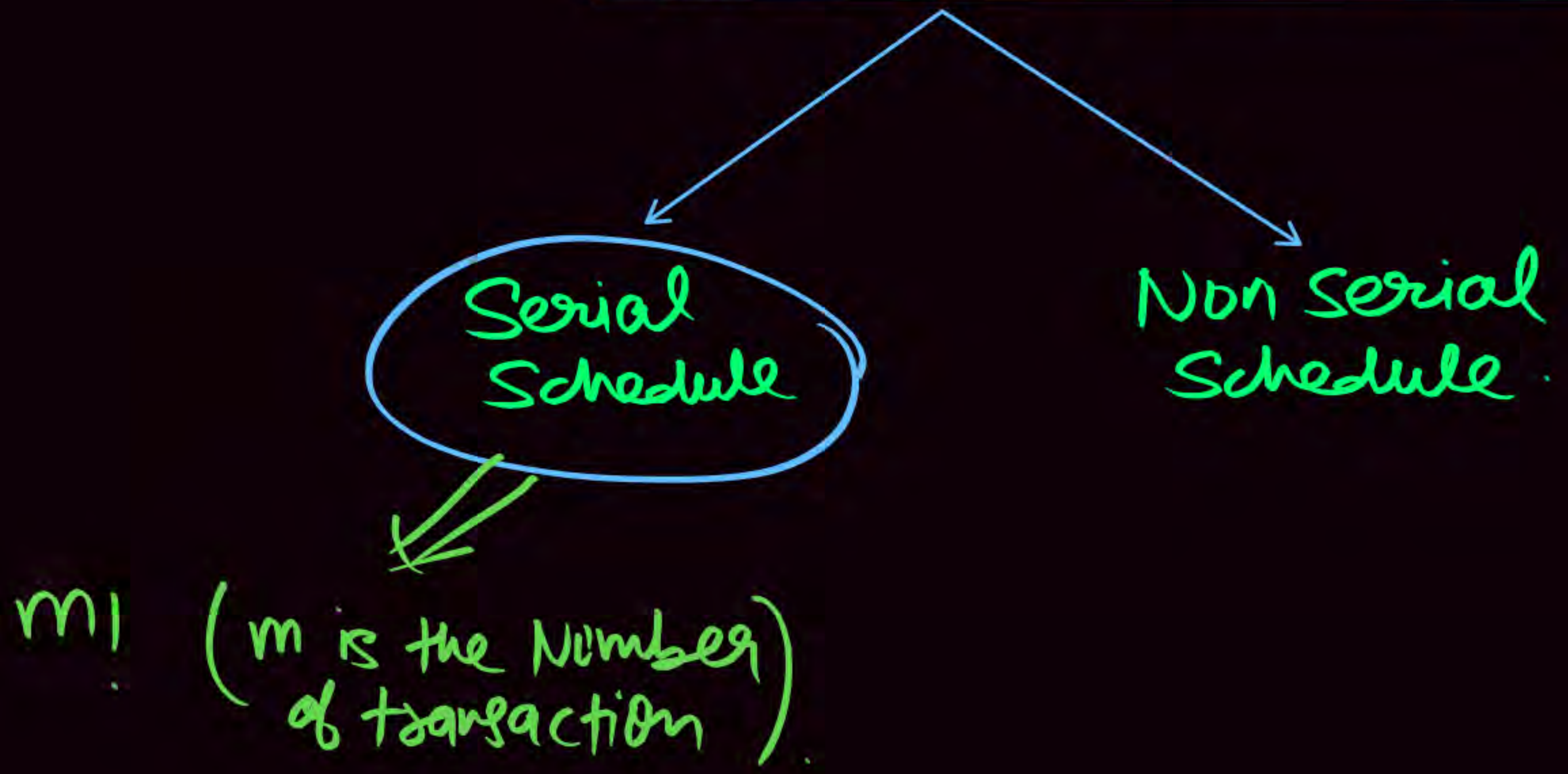
A Atomicity

[✓ B Consistency]

C Isolation

D Durability

Type of schedule



T_1 - 2 operation

T_2 ⇒ 2 operation .

T_1	T_2
r(A)	r(B)
w(A)	w(B)

T_1	T_2
r(A)	
w(A)	
	r(B)
	w(B)

S_1<T_1 T_2>

1

T_1	T_2
	r(B)
	w(B)
r(A)	
w(A)	

S_2<T_2 T_1>

2

Serial

T_1	T_2
r(A)	
	r(B)
w(A)	
	w(B)

3

T_1	T_2
r(A)	
	r(B)
	w(B)
w(A)	

4

T_1	T_2
	r(B)
r(A)	
	w(B)
w(A)	

5

T_1	T_2
	r(B)
r(A)	
w(A)	
	w(B)

6

2 Serial
+4 Non Serial

Total 6 Concurrent

Px	Py
A	I
B	II

Serially
AB I II
II A B

Non serial
A I B II or A I II B
I A II B or I A B II

Px	Py
L_1	L_3
L_2	L_4

serial
$L_1 L_2$ $L_3 L_4$
$L_3 L_4$ $L_1 L_2$

$L_1 L_3 L_2 L_4$ or $L_1 L_3 L_4 L_2$
$L_3 L_1 L_4 L_2$ or $L_3 L_1 L_2 L_4$

Px	Py
0	1
0	1

0011
1100
0101 or 0110
1010 or 1001

$T_1 \rightarrow 2$ operation (n_1)

$T_2 \rightarrow 2$ operation (n_2)

m: # of transaction

Total Number of Concurrent Schedule $= \dfrac{(n_1+n_2)!}{(n_1)!\,(n_2)!} \Rightarrow \dfrac{(2+2)!}{(2!)\,(2!)}$

$$= \frac{4!}{2\times2} = \frac{\cancel{4}\times3\times2}{\cancel{2\times2}} = 6$$

Total Concurrent Schedule $= 6$

Serial Schedule - $m! \Rightarrow 2! =$ 2 serial schedule

Non Serial = Concurrent − serial

$= 6 - 2$

$= 4$ Ans

If $T_1, T_2, T_3 \ldots T_m$ Transaction having $n_1, n_2, n_3 \ldots n_m$ operation Respectively.

Total Number of Concurrent Schedule $= \dfrac{(n_1 + n_2 + n_3 + \ldots n_m)!}{(n_1)!\,(n_2)!\,(n_3)! \ldots (n_m)!}$

Total Number of Serial Schedule $= m!$ (m : # of transaction)

Total Non Serial = Total Concurrent — Total Serial

eg $T_1 \rightarrow 1$ operation
$T_2 \rightarrow 2$ operation
$T_3 \rightarrow 3$ operation

Q. (i) Total Concurrent ?

Q.(ii) Total serial ?

Q(iii) Non serial ?

Soln Total Number of Concurrent $= \frac{(1+2+3)!}{(1)!\,(2)!\,(3)!} = \frac{6!}{(2!)\,(3!)} \Rightarrow \frac{\overset{3}{\cancel{6}}\times 5\times 4\times \cancel{3!}}{\cancel{2}\times \cancel{3!}}$

$= \boxed{60}$

Serial Schedule $= 3! = 6$ Serial Schedule

Non Serial $= 60 - 6 = 54$ Ans

Any Doubt ?

THANK YOU!