*** classification of schedules based on serializability:-

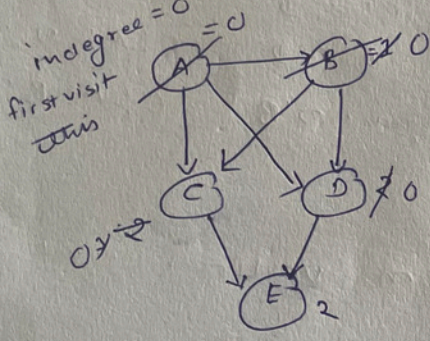
a) Conflict Serializable schedule:- ⇒ ① Topological order
(To check whether the given concurrent schedule is conflict serializable or not.) ② conflict pair
③ precedence graph.
④ conflict equal schedules:-

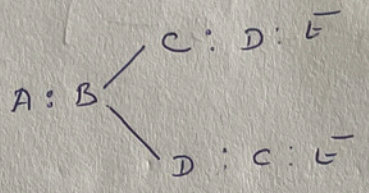
* Topological order:- (Graph traversal Algo. which is used for only direct acyclic Graph)

Step ① visit vertex (v) with indegree "0" and delete visited vertex (v) from Graph.

Step ② Repeat ① until every vertex visited.

⇒ Find the topological order of this given graph:-

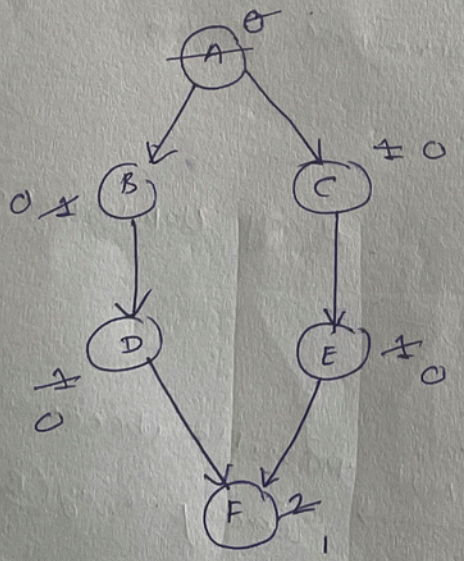


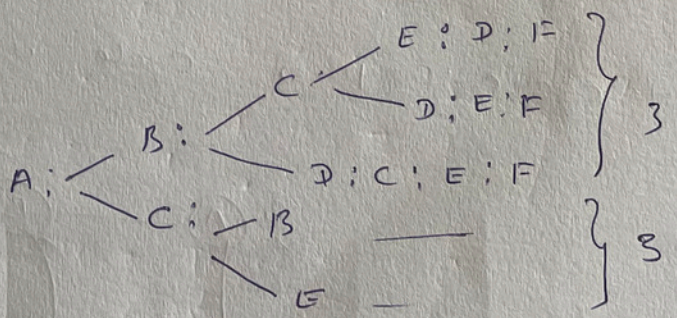


So 2 Topological orders are possible:-

→ A : B : C : D : E

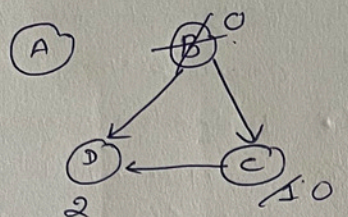
→ A : B : D : C : E





3 + 3 ≠ 6 Topological order.

⇒



① B ; C ; D : A

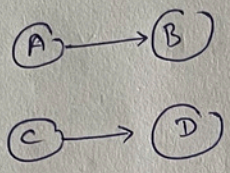
② A : B ; C : D

③ B : A : C : D

④ B : C : A : D :

Total NO. = 4.

⇒



A : B

C : D

[A — B —] ⇒ A ; B ; C ; D
C : D : A : D
A ; C : D : B

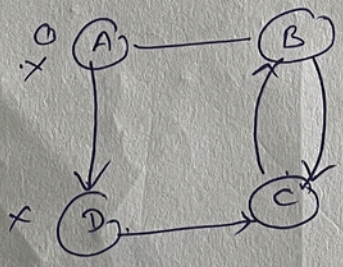
A : C : B : D
C ; A ; D : A
C ; A ; B : D

} = 6

* NULL Graph : Graph with n vertices and 0 edges

topological order = L^n.

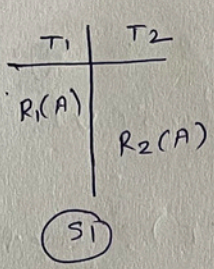
* cyclic Graph ⇒ Cycle in Directed graph.

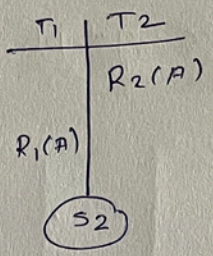


# of topological order = 0

A : D : ⟶ Not possible because indegree of (B) and (C) never zero.

* Conflict pair :=



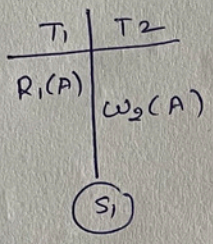


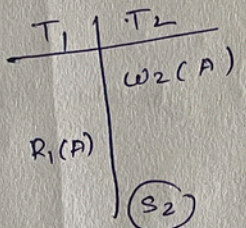
result of S_1 and S_2 are equal

$\Rightarrow$ $S_1 = S_2$

Non - conflict pair :-

$\Rightarrow$





$S_1 \neq S_2$

Conflict pair

$\Rightarrow$ Find the conflict pair :$\Rightarrow$

(1) S : $r_1(A)$ — $r_2(A)$

(2) S : $r_1(A)/w_1(A)$ — — — $r_2(B)/w_2(B)$

} Non - conflict pair.

(3) S : $r_1(A)$ — — $w_2(A)$

(4) S : $w_1(A)$ — — $r_2(A)$

(5) S : $w_1(A)$ — — $w_2(A)$

} conflict pair

$\Rightarrow$ Conflict pair : two operations conflict only if

(a) At least one write

and

(b) Same data item

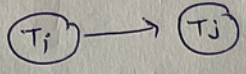
and

(c) both are from different transaction.

→ precedence graph:- This is a graph G(V, E).

Vertices (V): Transactions of the schedule.

Edges (E): conflict pair procedences of given schedule.

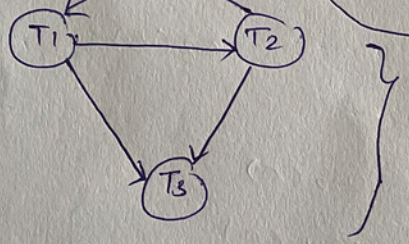


S: $r_i(A)$ — $w_j(A)$

S: $w_i(A)$ — $R_j(A)$ / $w_j(A)$.

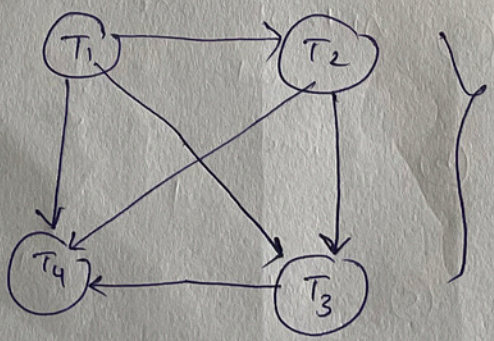
→ construct precedence graph for the following schedule:-

(1) S: $r_1(A)$ $r_2(B)$ $w_2(A)$ $w_2(B)$ $w_1(A)$ $r_3(B)$ $w_3(A)$



precedence graph of this schedule.

(2). S: $r_1(A)$ $r_2(A)$ $r_3(A)$ $r_4(A)$ $w_1(B)$ $w_2(B)$ $w_3(B)$ $w_4(B)$

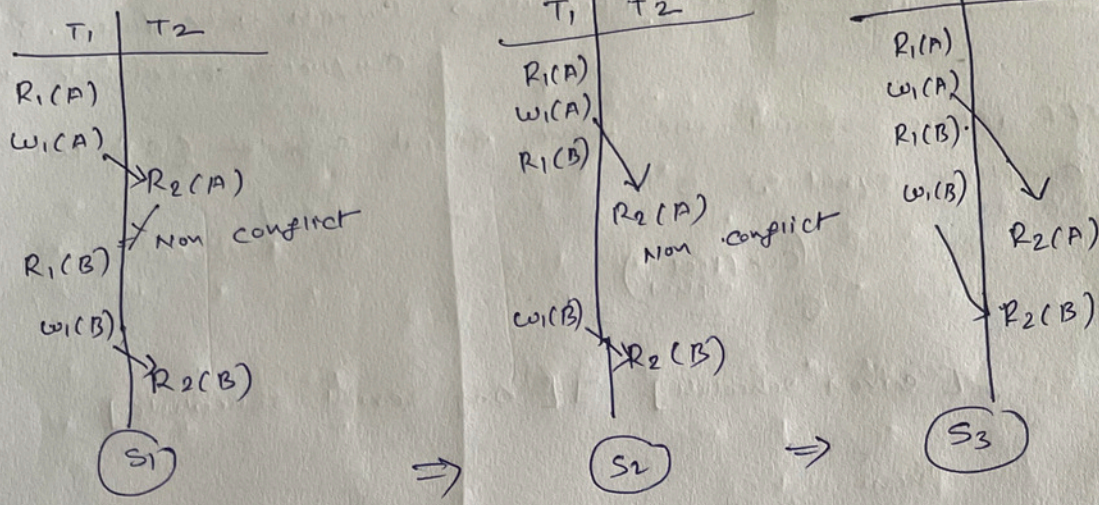


precedence graph.

Conflict Equal Schedule:- S_1 & S_2 conflict equal schedules Iff some consecutive non-conflict pair of S_1 exchanged which is S_2

$S_1 \Rightarrow S_2$

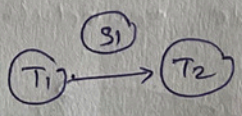
Ex.

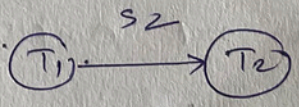


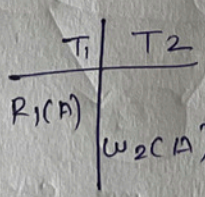
⇒ S_1, S_2 and S_3 are conflict equal schedule.

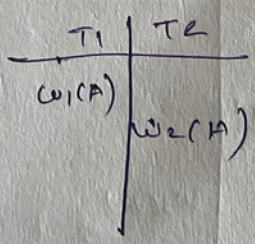
$S_1 \equiv T_1 \rightarrow T_2$, $S_2 \equiv T_1 \rightarrow T_2$, $T_1 \rightarrow T_2$

* ⇒ S_1, S_2, and S_3 are conflict equal they their precedence graph the same. but reverse of this statement is not right. For second may / may not condition follows:-









So, S_1 and S_2 are not Conflict equal.

* If transaction of S_1 and S_2 schedules are same and Precedence graphs of S_1, S_2 are same then S_1, S_2 are conflict equal schedule.

* If S_1 & S_2 are conflict equal then result of S_1 and S_2 is same.

* Conflict serializable schedule ⇒ Given schedule .S is conflict serializable schedule

Iff there should be some serial schedule (S') which is conflict equal to S.

$$S \overset{\text{(conflict equal)}}{=} S'$$

[Given schedule] ∃ [Any serial schedule]

Then "S" is conflict serializable schedule.

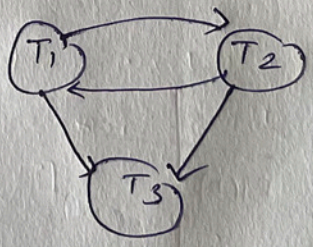
⇒ S is conflict serializable schedule iff and only if after exchanging of consecutive non-conflict pairs of (S) should result any one serial schedule (S').

⇒ Testing of conflict serializability schedule ⇒ If

(i) Precedence graph of the given schedule is cyclic then S is not conflict serializable schedule.

Ex:-

T_1	T_2	T_3
$R_1(A)$		
	$W_2(A)$	
$W_1(A)$		
		$W_3(A)$



here generated cycle.

So it is not conflict serializable schedule.

[None of the serial (S') schedule conflict equal to given schedule.]

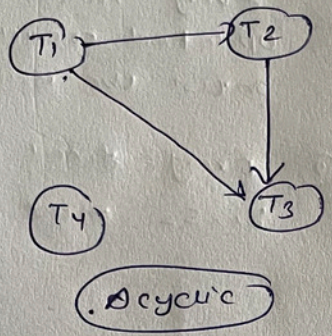
* (2) If precedence graph of the given schedule is acyclic then given schedule is conflict serializable schedule equal serial schedule of given schedule is topological orders of precedence graph.

Ex: S: $r_1(A)$ $r_2(A)$ $r_3(A)$ $r_4(A)$ $w_1(B)$ $w_2(B)$ $w_3(B)$ $w_4(D)$.

(a) test S is conflict serializable schedule or not

(b) If conflict serial schedule then find # of conflict equal serial schedule.

(a)



Acyclic

(a) This is conflict serializable schedule.

(b) Topological order : $T_1\ T_2\ T_3$

$T_1\ T_2\ T_3\ T_4$

$T_1\ T_4\ T_2\ T_3$

$T_4\ T_1\ T_2\ T_3$

$T_1\ T_2\ T_4\ T_3$ } 4

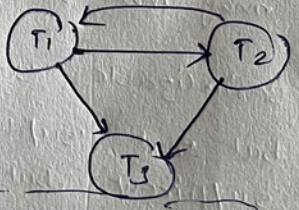
serial schedule(4)

these are conflict equal to serial schedule.

Ex:

T_1	T_2	T_3
$R_1(A)$		
	$W_2(A)$	
$W_1(A)$		
		$W_3(A)$

Solve:- Precedence graph:



S: is not conflict serializable schedule.

[No serial schedule is conflict equal to S]

⇒ S': $T_1 \rightarrow T_2 \rightarrow T_3$ (serial order)

T_1	T_2	T_3
$R(A)$		
$W_1(A)$		
	$W_2(A)$	
		$W_3(A)$

So S & S' is not conflict equal but S & S' is view Equal schedule.

⇒ S_1 & S_2 Conflict equal

① Every R-W conflict pair & W-R conflict pair of S_1 & S_2 must be same precedence

② Every W-W conflict pair of S_1 & S_2 must be same precedence

S_1 & S_2 view Equal

① ~~S_1 & S_2 view Equal.~~ Every initial read and updated read of S_1, S_2 must be same.

② Every final writes of S_1, S_2 must be same.

* If schedule S_1 and S_2 are conflict equal then S_1 and S_2 are also view equal.

* If S_1 and S_2 are not conflict equal then S_1 and S_2 may or may not view equal,

* If schedule S is conflict serializable schedule then S is also view serializable schedule.

* If schedule S is not conflict serializable schedule then S may (or) may not view serializable schedule

* Conflict serializable schedule testing condition only sufficient condition but not sufficient condition for serializability testing [only sufficient but not necessary]

or

[or only if]

0 cycle precedence group exists for S → S: serializable schedule

[T] → [T]
[F] → [T/F] To avoid this disadvantage we uses view serializable schedule testing

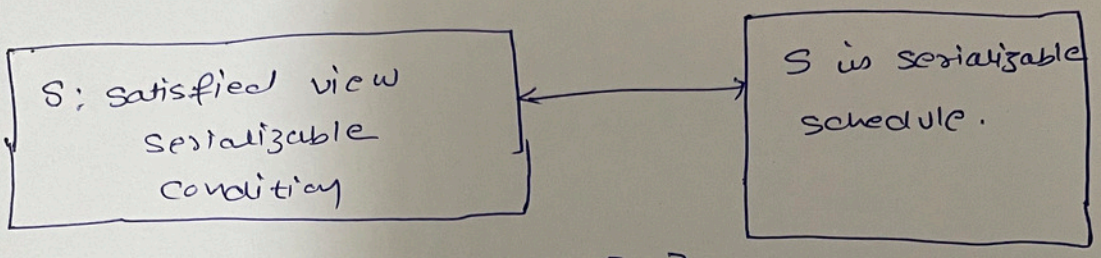
View serializability schedule testing conditions are sufficient and necessary conditions for serializable testing.

both sufficient and necessary

(or)

(or)

if and only if



[T] ⟷ [T]

[F] ⟷ [F].

view serializable schedule:-