Conflict Serializability Examples

Conflict Serializable: A schedule is called conflict serializable if it can be transformed into a serial schedule by swapping non-conflicting operations.

Conflicting operations: Two operations are said to be conflicting if all conditions satisfy:

- They belong to different transactions
- They operate on the same data item
- At Least one of them is a write operation

T2

		READ	WRITE
T1	READ	Allowed	Not allowed
	WRITE	Not allowed	Not allowed

Precedence Graph For Testing Conflict Serializability

Precedence Graph or **Serialization Graph** is used commonly to test Conflict Serializability of a schedule.
It is a directed Graph (V, E) consisting of a set of nodes V = {T1, T2, T3……….Tn} and a set of directed edges E = {e1, e2, e3………………em}.

Algorithm:

1.Create a node T in the graph for each participating transaction in the schedule.

2.For the conflicting operation read_item(X) and write_item(X) – If a Transaction Tj executes a read_item (X) after Ti executes a write_item (X), draw an edge from Ti to Tj in the graph.

3.For the conflicting operation write_item(X) and read_item(X) – If a Transaction Tj executes a write_item (X) after Ti executes a read_item (X), draw an edge from Ti to Tj in the graph.

4.For the conflicting operation write_item(X) and write_item(X) – If a Transaction Tj executes a write_item (X) after Ti executes a write_item (X), draw an edge from Ti to Tj in the graph.

5.The Schedule S is serializable if there is no cycle in the precedence graph.

Example 1:

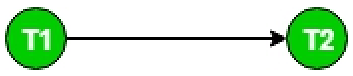
S : r1(x) r1(y) w2(x) w1(x) r2(y)

T1	T2
READ(X)	
READ(Y)	
	WRITE(X)
WRITE(X)	
	READ(Y)

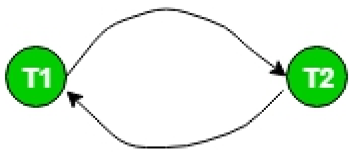
Creating Precedence graph:

1.Make two nodes corresponding to Transaction T1 and T2.

2.For the conflicting pair r1(x) w2(x), where r1(x) happens before w2(x), draw an edge from T1 to T2.



3.For the conflicting pair w2(x) w1(x), where w2(x) happens before w1(x), draw an edge from T2 to T1.

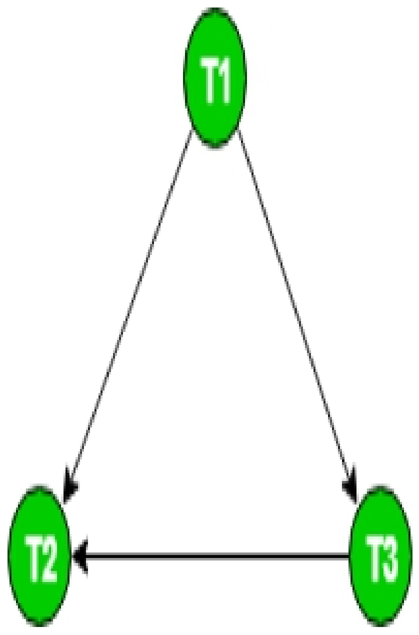


Since the graph is cyclic, we can conclude that it is **not conflict serializable** to any schedule serial schedule.

Example 2:

S1: r1(x) r3(y) w1(x) w2(y) r3(x) w2(x)

The graph for this schedule is :



Since the graph is acyclic, the schedule is conflict serializable.

Topological Sort on this graph would give us a possible serial schedule which is conflict equivalent to schedule S1.
In Topological Sort, we first select the node with indegree 0, which is T1. This would be followed by T3 and T2.
So,**S1 is conflict serializable** since it is **conflict equivalent** to the **serial schedule T1 T3 T2.**

Example 3:

S : r1(X), w1 (X), r2 (X), r3(X), w3(X), r3(X), r1(X), w2(X), r4(X), r2(X), r3(X), w3(X)