Database Management System (DBMS)

Lecture-40

Dharmendra Kumar

December 15, 2020

View Serializability

View equivalent: Consider two schedules S and S', where the same set of transactions participates in both schedules. The schedules S and S' are said to be view equivalent if following three conditions are satisfied:
1. For each data item Q, if transaction T_i reads the initial value of Q in schedule S, then transaction T_i must, in schedule S', also read the initial value of Q.
2. For each data item Q, if transaction T_i executes read(Q) in schedule S, and if that value was produced by a write(Q) operation executed by transaction T_j , then the read(Q) operation of transaction T_i must, in schedule S', also read the value of Q that was produced by the same write(Q) operation of transaction T_j .

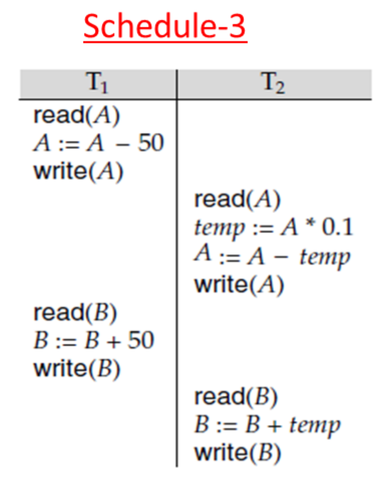
Transaction

3. For each data item Q, the transaction (if any) that performs the final write(Q) operation in schedule S must perform the final write(Q) operation in schedule S'.

View serializable: A schedule S is said to be view serizaliabe if it is view equivalent to a serial schedule.

Transaction

Example: Consider the following schedule-3:-



Schedule-3

T_1	T_2
read(A)	
$A := A - 50$	
write(A)	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
read(B)	
$B := B + 50$	
write(B)	
	read(B)
	$B := B + temp$
	write(B)

Is this schedule view serializable?

Transaction

Solution:

First, we check the view equivalent of schedule-3 with serial schedule-1 i.e. $T_1 T_2$.

First consider data item A.

Clearly, in schedule-3 and schedule-1, transaction T_1 reads the initial value of A, therefore condition-1 is satisfied for data item A.

Clearly, in schedule-3, transaction T_2 reads the value of A written by T_1. In schedule-1, transaction T_2 reads the value of A written by T_1. Clearly same order of write-read occur in both schedule, therefore second condition is also satisfied.

Clearly, in schedule-3, final write(A) operation is performed by transaction T_2. In schedule-1, final write(A) operation is also performed by transaction T_2. Therefore, third condition is also satisfied.

Clearly all the three conditions are satisfied for data item A.

Now, we will check all the three conditions for data item B.

Transaction

Clearly, in schedule-3 and schedule-1, transaction T_1 reads the initial value of B, therefore condition-1 is satisfied for data item B.

Clearly, in schedule-3, transaction T_2 reads the value of B written by T_1. In schedule-1, transaction T_2 reads the value of B written by T_1. Clearly same order of write-read occur in both schedule, therefore second condition is also satisfied.

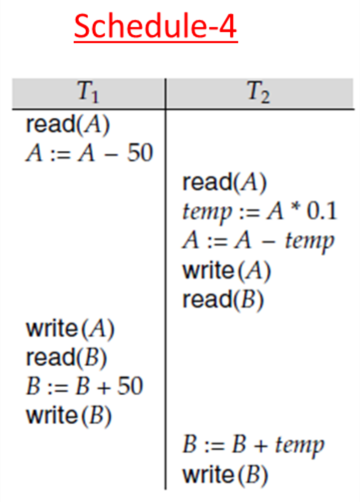
Clearly, in schedule-3, final write(B) operation is performed by transaction T_2. In schedule-1, final write(B) operation is also performed by transaction T_2. Therefore, third condition is also satisfied.

Clearly all the three conditions are satisfied for data item B.

Since all the three conditions are satisfied for each data item in both schedules 1 and 3, therefore both schedule-3 is view equivalent to serial schedule-1. Therefore, schedule-3 is view serializable.

Transaction

Example: Consider the following schedule-4:-



Schedule-4

T_1	T_2
read(A)	
$A := A - 50$	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
	read(B)
write(A)	
read(B)	
$B := B + 50$	
write(B)	
	$B := B + temp$
	write(B)

Is this schedule view serializable?

Transaction

Solution:

First, we check the view equivalent of schedule-4 with serial schedule-1 i.e. $T_1 T_2$.

First consider data item A.

Clearly, in schedule-4 and schedule-1, transaction T_1 reads the initial value of A, therefore condition-1 is satisfied for data item A.

Clearly, in schedule-4, there is no transaction which reads the value of A written by any transaction. Therefore second condition is also satisfied.

Clearly, in schedule-4, final write(A) operation is performed by transaction T_1. But, in schedule-1, final write(A) operation is performed by transaction T_2. Therefore, third condition is not satisfied.

Therefore, schedule-4 is not view equivalent to a serial schedule $T_1 T_2$.

Now, we check the view equivalent of schedule-4 with serial schedule-2 i.e. $T_2 T_1$.

First consider data item A.

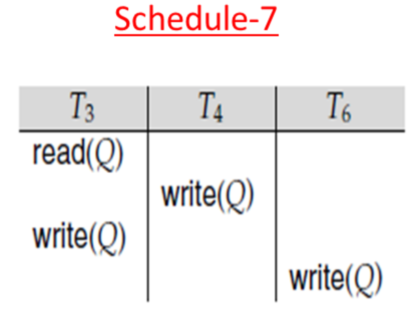
Clearly, in schedule-4, transaction T_1 reads the initial value of A, but in serial schedule-2, transaction T_2 reads the initial value of A. Therefore condition-1 is not satisfied for data item A.

Therefore, schedule-4 is not view equivalent to a serial schedule $T_1 T_2$.

Therefore, schedule-4 is not view serializable.

Transaction

Example: Consider the following schedule-7:-



Schedule-7

T_3	T_4	T_6
read(Q)		
	write(Q)	
write(Q)		
		write(Q)

Is this schedule view serializable?

Solution:

Clearly this schedule is view equivalent to serial schedule $T_3 T_4 T_6$. Therefore, this schedule is view serializable.

Testing for Serializability

In this section we are going to study a simple and efficient method for determining conflict serializability of a schedule. This method is explained as following.

Consider a schedule S. We construct a directed graph, called a **precedence graph**, from S. This graph consists of a pair G = (V, E), where V is a set of vertices and E is a set of edges. The set of vertices consists of all the transactions participating in the schedule. The set of edges consists of all edges $T_i \rightarrow T_j$ for which one of three conditions holds:

1. T_i executes write(Q) before T_j executes read(Q).
2. T_i executes read(Q) before T_j executes write(Q).
3. T_i executes write(Q) before T_j executes write(Q).

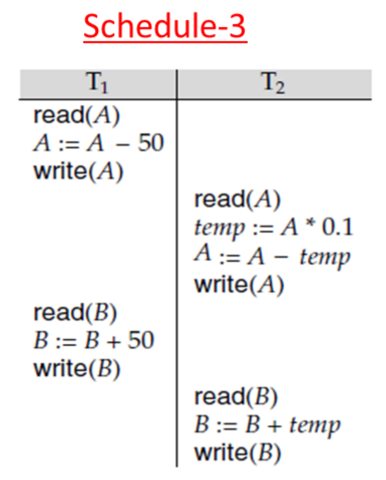
Transaction

If the precedence graph for S has a cycle, then schedule S is not conflict serializable. If the graph contains no cycles, then the schedule S is conflict serializable.

A **serializability order** of the transactions can be obtained through topological sorting, which determines a linear order consistent with the partial order of the precedence graph.

Transaction

Example: Consider the following schedule-3:-



Schedule-3

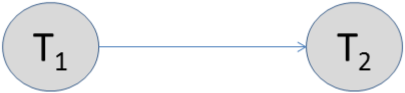
T_1	T_2
read(A)	
$A := A - 50$	
write(A)	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
read(B)	
$B := B + 50$	
write(B)	
	read(B)
	$B := B + temp$
	write(B)

Construct precedence graph for it.

Transaction

Solution:

Precedence graph for above schedule will be the following:-

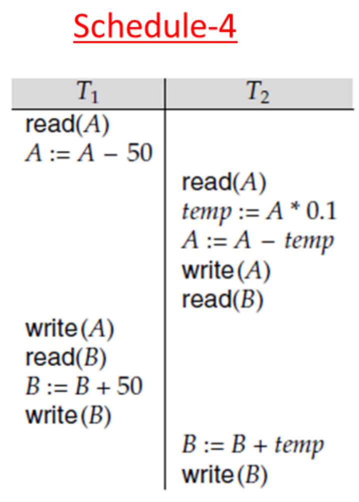


Precedence graph for schedule-3

Clearly this graph does not contain any cycle. Therefore, this schedule is conflict serializable.

Transaction

Example: Consider the following schedule-4:-



Schedule-4

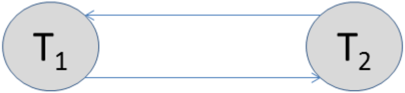
T_1	T_2
read(A)	
$A := A - 50$	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
	read(B)
write(A)	
read(B)	
$B := B + 50$	
write(B)	
	$B := B + temp$
	write(B)

Construct precedence graph for it.

Transaction

Solution:

Precedence graph for above schedule will be the following:-

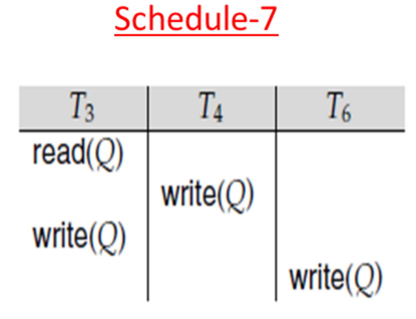


Precedence graph for schedule-4

Clearly this graph contains a cycle. Therefore, this schedule is not conflict serializable.

Transaction

Example: Consider the following schedule-7:-



Schedule-7

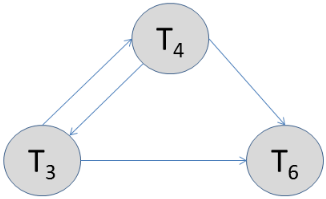
T_3	T_4	T_6
read(Q)		
	write(Q)	
write(Q)		
		write(Q)

Construct precedence graph for it.

Transaction

Solution:

Precedence graph for above schedule will be the following:-



Precedence graph for schedule-7

Clearly this graph contains a cycle. Therefore, this schedule is not conflict serializable.

Note: If a schedule is conflict serializable then it is also view serializable. But converse need not be true.