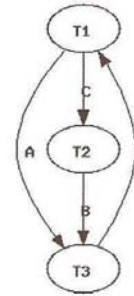


1. Read and Read/Write Locks

Time	T1	T2	T3	Conflict	Edge
1	RL A				
2		RL B			
3	RL C				
4	UL A				
5			RWL A	1: T1 (RL A)	$T1 \xrightarrow{A} T3$
6			UL A		
7		UL B			
8	RWL A			5: T3 (RWL A)	$T3 \xrightarrow{A} T1$
9	UL C				
10		RWL C		3: T1 (RL C)	$T1 \xrightarrow{C} T2$
11			RL B	No conflict	
12			UL B		
13	UL A				
14			RWL B	2: T2 (RL B)	$T2 \xrightarrow{B} T3$
15		UL C			
16			UL B		

Conflict graph:

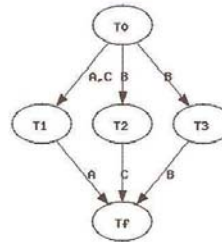


This schedule is not serializable because of the cycle in the graph between T1 and T3 over attribute A. (Topological sort immediately fails, as no nodes are free from incoming edges).

2. Read and Write Locks

Time	T1	T2	T3	Read Edges
0	T0 writes A,B,C			
1	RL A			$T0 \xrightarrow{A} T1$
2		RL B		$T0 \xrightarrow{B} T2$
3	RL C			$T0 \xrightarrow{C} T1$
4	UL A			
5			WL A	
6			UL A	
7		UL B		
8	WL A			
9	UL C			
10		WL C		
11			RL B	$T0 \xrightarrow{B} T3$
12			UL B	
13	UL A			
14			WL B	
15		UL C		
16			UL B	
f		Tf reads A		$T1 \xrightarrow{A} Tf$
f		Tf reads B		$T3 \xrightarrow{B} Tf$
f		Tf reads C		$T2 \xrightarrow{C} Tf$

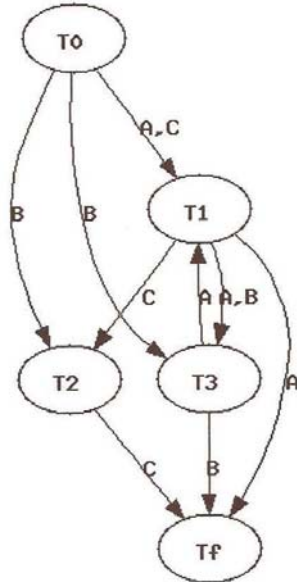
Initial graph:



There are no useless transactions to remove. Add edges and/or edge pairs:

Initial Edges	Edges Added
$T0 \xrightarrow{A} T1$	$T1 \xrightarrow{A} T3$
$T0 \xrightarrow{B} T2$	$T2 \xrightarrow{B} T3$
$T0 \xrightarrow{C} T1$	$T1 \xrightarrow{C} T2$
$T0 \xrightarrow{B} T3$	No other T writes B
$T1 \xrightarrow{A} Tf$	$T3 \xrightarrow{A} T1$
$T3 \xrightarrow{B} Tf$	No other T writes B
$T2 \xrightarrow{C} Tf$	No other T writes C

No edge pairs were added, so we just have a simple, single graph again:



Since there were no edge pairs, we only need to check this one graph to determine if it is acyclic and therefore serializable. The schedule is not serializable because of the cycle that still exists between T1 and T3 over attribute A. Topological sort will remove node T0 and all of its edges but will then stall as every other node has an incoming edge:

