First the graph can ran all transaction in T1, then T3, and finally T2 with the same output, so the transaction is serializable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T1 | T2 | T3 | Precedence Graph: | Conflict |
| R(A) |  |  | T1 connect T3 | Still conflict |
| W(A) |  |  | T1 connect T2, T1 connect T3 | Still conflict |
|  |  | R(A) |  | Still conflict |
|  |  | W(A) | T3 connect T2 | Still conflict |
|  | R(A) |  |  | Still conflict |
| R(B) |  |  | T1 connect T3 | Still conflict |
|  |  | R(B) | T3 connect T1 | RULE BREAK |
| W(B) |  |  |  |  |
|  |  | W(B) |  |  |
|  | R(B) |  |  |  |
|  | commit |  |  |  |
| commit |  |  |  |  |
|  |  | commit |  |  |

So, NO, it’s not conflict serializable, and because there is a circle forms (t1 to t3 and t3 to t1). Also, it’s also because we cannot change the order of 7th and the 8th transaction, which will result in different result.