Proof of work

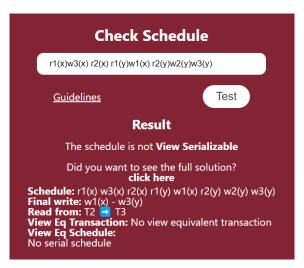
Now I am testing the code on some exercises from the exam, checking whether the solutions proposed by my program are equal to the ones provided in the exam.

1) r1(x)w3(x) r2(x) r1(y)w1(x) r2(y)w2(y)w3(y)

This exercise is from 12/07/2019, and the solution says:

Since $w_3(y)$ is the final write on y in S, and $w_2(y)$ is in S, we observe that T_3 must appear after T_2 in every serial schedule S' on T_1, T_2, T_3 that is view-equivalent to S, because otherwise S' would have a different "final write se" with respect to S. On the other hand, the sequence $w_3(x) r_2(x)$ in S implies that in any serial schedule S' on T_1, T_2, T_3 that is view-equivalent to S transaction T_2 appears immediately after transaction T_3 , because otherwise S' would have a different "read from" relation with respect to S. We conclude that no serial schedule on T_1, T_2, T_3 exists that is view-equivalent to S, i.e., S is not view-serializable.

VS Playground solution:



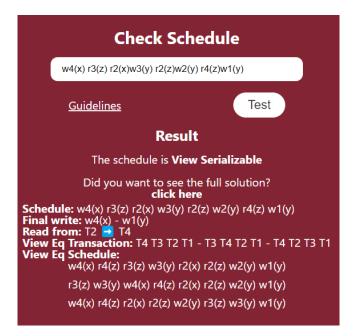
2) w4(x) r3(z) r2(x)w3(y) r2(z)w2(y) r4(z)w1(y)

01/02/2016, solution:

• The serial schedule T_4, T_2, T_3, T_1 has the same read-from relation and the same final-write-set as S, whereas it has the conflicting action pair $(w_2(y), w_3(y))$ that appears in different order with respect to S. Therefore, such serial schedule is view equivalent but not conflict equivalent to S.

The exercise requires the user to identify a view-equivalent schedule that is not conflict-serializable. I need to verify whether the serial schedule proposed in the solution matches the one I provided.

VS Playground solution:

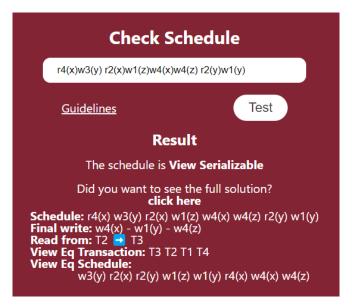


3) r4(x)w3(y) r2(x)w1(z)w4(x)w4(z) r2(y)w1(y)

21/2/2014, solution:

S is view-serializable, since the serial schedule T_3, T_2, T_1, T_4 is clearly view-equivalent to S, because it has the same read-from relation and the same final-set as S. There is a single action that can be added to S in such a way that the resulting schedule is no longer view serializable: for example, if we add $r_1(x)$ at the end of the schedule, then the resulting schedule S' is no longer view serializable, because now T_1 reads from T_4 (implying that in every serial schedule view equivalent to S', T_4 preceeds T_1), and in S the final write on S is S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S', S implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent to S' implying that in every serial schedule view equivalent view eq

VS Playground solution:



4) r1(A) r3(C) w3(B) r2(A) w1(B) w1(A) w2(A) r1(C) w3(C) r3(A) r2(D)

18/02/2010, solution:

S is not view-serializable. Indeed, every serial schedule S' on the set of transactions constituting S will have either transaction t_1 before t_2 , or t_2 before t_1 . In the former case (t_1 before t_2), the pair $\langle w_1(A), r_2(A) \rangle$ is in the READS-FROM relation associated to S', while it is not in the READS-FROM relation associated to S. In the latter case (t_2 before t_1), the pair $\langle w_2(A), r_1(A) \rangle$ is in the READS-FROM relation associated to S', while it is not in the READS-FROM relation associated to S. Actually, it is easy to see that S is not even conflict-serializable, because the precedence graph associated to S is cyclic.

VS Playground solution:

