Abhinav Yalamaddi - YXA210040

Database Design - CS 6360.002 - Fall 2022 - HW3B

21.9. What is a serial schedule? What is a serializable schedule? Why is a serial schedule considered correct? Why is a serializable schedule considered correct?
Theoretically, a schedule S is serial if all of the activities of each transaction T participating in the schedule are executed sequentially in the schedule; otherwise, the schedule is nonserial.
A serializable schedule is defined as follows: A serializable schedule S of n transactions is one that is comparable to some serial schedule with the same n transactions.
If we assume the transactions to be independent, one fair assumption is that every serial schedule is correct. We may presume this since each transaction is deemed accurate if performed independently. As a result, it makes no difference whether the transaction is processed first. We receive a valid final result in the database as long as each transaction is completed from start to finish in isolation from the activities of other transactions.
It is regarded as accurate since a serial timetable is identical to a serial schedule.

21.22. Which of the following schedules is (conflict) serializable? For each serializable schedule, determine the equivalent serial schedules.
a. r1(X); r3(X); w1(X); r2(X); w3(X);
Because T1 reads X (r1(X)) before T3, yet T3 reads X (r3 (X)) before T1 writes X (w1(X)), this schedule is **not serializable**. T2's operation r2 (X) has no effect on the timetable, hence its location in the schedule is meaningless. The operation w1 (X) arrives after r3 (X) in a serial schedule T1, T2, and T3, which does not happen in the query.

b. r1(X); r3(X); w3(X); w1(X); r2(X);
This schedule **cannot be serialized** because T1 reads X (r1 (X)) before T3, yet T3 writes X (w3 (X)) before T1. T2's operation r2(X) has no effect on the timetable, so its location in the schedule is meaningless. In a serial schedule T1, T3, and T2, r3 (X) and w3 (X) must follow w1 (X), which does not occur in the question.

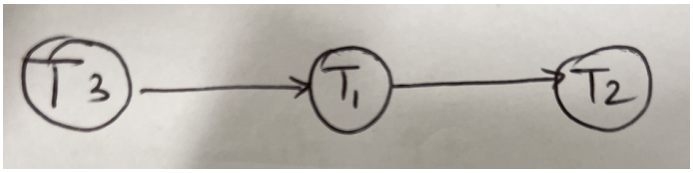
c. r3(X); r2(X); w3(X); r1(X); w1(X);
This schedule is **serializable** since all competing T3 operations occur before all conflicting T1 operations. T2 only has one operation, a read on X (r2 (X)), that does not clash with any other operation. As a result, this serializable schedule is identical to the serial schedules r2 (X); r3 (X); w3 (X); r1 (X); w1 (X).

d. r3(X); r2(X); r1(X); w3(X); w1(X);
This is **not** a **serializable** schedule because T3 reads X (r3 (X)) before T1 reads X (r1 (X)), yet T3 writes X (w3 (X)) before T3. In a serial schedule with T3, T2, and T1, r1 (X) occurs after w3 (X), which does not occur in the question.

Abhinav Yalamaddi
YXA210040

21.23. Consider the three transactions T1, T2, and T3, and the schedules S1 and S2 given below. Draw the serializability (precedence) graphs for S1 and S2, and state whether each schedule is serializable or not. If a schedule is serializable, write down the equivalent serial schedule(s).
T1: r1 (X); r1 (Z); w1 (X);
T2: r2 (Z); r2 (Y); w2 (Z); w2 (Y);
T3: r3 (X); r3 (Y); w3 (Y);
S1: r1 (X); r2 (Z); r1 (Z); r3 (X); r3 (Y); w1 (X); w3 (Y); r2 (Y); w2 (Z); w2 (Y);
S2: r1 (X); r2 (Z); r3 (X); r1 (Z); r2 (Y); r3 (Y); w1 (X); w2 (Z); w3 (Y); w2 (Y);

Schedule S1: It is a serializable schedule because T1 only reads X (r1(X)), which is not modified either by T2 or T3. T3 reads X (r3 (X)) before T1 modifies it (w1 (X)), and T2 reads Y (r2 (Y)) and writes it (w2 (Y)) only after T3 has written to it (w3 (Y)).



Schedule S2: It is not a searilizable schedule because T2 reads Y (r2 (Y)), which is then read and modified by T3 (w3 (Y)), and T3 reads Y (r3 (Y)), which is then modified before T2 modifies Y (w2 (Y)).

In the above order, T3 interferes with the execution of T2, which makes the schedule non searializable.