

Database Tuning, ITU, Spring 2008

Rasmus Pagh

March 13, 2008

1 2PL and full locking together with read committed

We have considered concurrency control mechanisms based on two-phase locking (2PL). Transactions at lower isolation levels (e.g., `read committed`) may not use 2PL, but instead request and release locks before and after each statement.

a) Suppose that we have two concurrent transactions T_1 and T_2 , where T_1 uses 2PL, and T_2 is *read-only* and locks database elements only for the duration of single statements. Argue that the result of running T_1 and T_2 (considering database updates as well as results returned by transactions) may sometimes *not* be equivalent to a serial schedule.

Suppose that a DBMS implements *full locking*, meaning that all locks are obtained at the beginning of the transaction, and released at the end of the transaction.

b) Again, suppose that we have two concurrent transactions T_1 and T_2 . Now T_1 executes a single SQL statement and uses full locking, whereas T_2 is again *read-only* and locks database elements only for the duration of single statements. Argue that the result of running T_1 and T_2 is *always* equivalent to a serial schedule.