Lecturer: George Mertzios

Department of Computer Science
Durham University

Networks and Systems – Databases

Practical 3: Transactions and Concurrency

Question 1.

Consider the following two transactions T1 and T2:

T1	T2
read_item(X);	read_item(X);
X := X – N;	X := X+M;
write_item(X);	write_item(X)
read_item(Y);	
Y := Y+N;	
write_item(Y);	

The transactions can be written as follows using shorthand notation, where we replace "read" by "r" and "write" by "w":

T1	T2
r1(X);	
w1(X);	r2(X);
r1(Y);	w2(X);
w1(Y);	

List all the possible schedules for the transactions T1 and T2. Determine which of them are conflict serializable and which are not.

(Hint: the number of possible schedules is: $\frac{(4+2)!}{4!\cdot 2!}=15$)

Question 2.

Consider schedules S_1, S_2, S_3 below. We denote by $r_i(j)$ the read-operation of transaction T_i on data item j. Similarly, we denote by $w_i(j)$ the write-operation of transaction T_i on data item j. Furthermore, we denote by c_i the commit-operation of transaction T_i.

S_1: r_1(X); r_2(Z); r_1(Z); r_3(X); r_3(Y); w_1(X); c_1; w_3(Y); c_3; r_2(Y); w_2(Z); w_2(Y); c_2;
S_2: r_1(X); r_2(Z); r_1(Z); r_3(X); r_3(Y); w_1(X); w_3(Y); r_2(Y); w_2(Z); w_2(Y); c_1; c_2; c_3;
S_3: r_1(X); r_2(Z); r_3(X); r_1(Z); r_2(Y); r_3(Y); w_1(X); c_1; w_2(Z); w_3(Y); w_2(Y); c_3; c_2;

(a) For each schedule, determine whether it is cascadeless.
(b) For each schedule, determine whether it is recoverable.

(c) For each schedule, draw its precedence (i.e. its conflict serialization) graph, and determine whether it is serializable. If so, provide an equivalent serial schedule; if not, explain why.

In each case, give sufficient reasons to explain your answer.

Question 3.

The definition of a schedule assumes that operations can be totally ordered by time. Consider a database system that runs on a system with multiple processors, where it is not always possible to establish an exact ordering between operations that are executed on different processors. However, operations on a data item can be totally ordered.

Does this situation cause any problem for the definition of conflict serializability? Explain your answer and give an example.

Question 4* (a bit more challenging).

Show that the two-phase locking (2PL) protocol ensures conflict serializability and that transactions can be serialized according to their lock points.