Exercises *Databases*
Session 5: concurrency

Hans Philippi, Lennart Herlaar, Ad Feelders

March 8, 2017

Exercise 26

In some situations, there are *procedures* to compensate for the real world effects of a committed transaction. Can you think of some examples?

Exercise 27

Give some examples of real world situations where there are severe problems if you want to undo the effects of a committed transaction.

Exercise 28

Think of a database system for a railroad company, managing the scheduling of people and materials.
Give at least two examples of things that can go (very) wrong if the system would not support concurrency control.

Exercise 29

We have two schedules:

S1

T1	T2	T3	T4	T5
				w(z)
	w(y)			
w(x)				
			r(z)	
		r(x)		
				w(x)
	w(z)			
		r(y)		

S2

T1	T2	T3	T4	T5
				w(z)
		r(y)		
w(x)				
			r(z)	
		r(x)		
				w(x)
	w(z)			
	w(y)			

Construct the precedence graphs. Determine the serializability of these schedules.
If a schedule is serializable, then give the equivalent serial schedule.

Exercise 30

Let us take a look again at the schedules of the previous exercise
Determine which of these schedules are allowed by a 2PL scheduler. Which conclusions can you draw?

Exercise 31

Suppose we use a time out mechanism to detect deadlock. What are the disadvantages of making the time out period long? What are the disadvantages of keeping the time out period short?

Exercise 32

Describe a method to prevent/detect deadlock based on *wait-for graphs.*

Exercise 33(!)

The theorem stating that serializability can be checked by analyzing the precedence graph of the schedule is based on the notion of a *topological sort* of a directed graph. A TopSort of a directed graph G is an enumeration of all the nodes in the graph, such that all the edges in the graph point 'from left to right' in the enumeration. $T_i \rightarrow T_j$ in G, then T_i occurs before T_j in Topsort($G(H)$).

(i) Give all TopSorts of the directed graph in figure 1

Graph theory guarantees us that a directed graph has a topological sort iff it is acyclic. This gives us the possibility to construct the equivalent serial schedule in an algoritmic way. Having a history H, we determine Topsort($G(H)$) and regard this as a serial history.

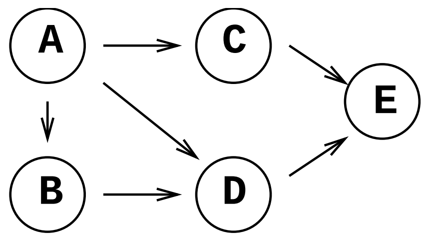

Figure 1: directed acyclic graph

(ii) Prove the following claim:
If history H has an acyclic precedence graph $G(H)$, then Topsort($G(H)$), regarded as a history, is equivalent to H.

Exercise 34(!)

We will give a proof that the 2PL mechanism only allows histories that are serializable. We define the peak moment p_i of a transaction T_i as a moment in time between the last lock operation of T_i and the first unlock operation of T_i. Note that the existence of such a moment is guaranteed by 2PL. Prove the following lemma with respect to the serialization graph.

If $T_i \rightarrow T_j$ then $p_i < p_j$

where $<$ denotes time order.
The final step is straightforward. Suppose that a 2PL scheduler allows a history with a cyclic serialization graph. Show that this leads to a contradiction.