Universität Freiburg
Institut für Informatik
Prof. Dr. G. Lausen
Alexander Schätzle
Martin Przyjaciel-Zablocki

Georges-Köhler Allee, Geb. 51
D-79110 Freiburg
lausen@informatik.uni-freiburg.de
schaetzl@informatik.uni-freiburg.de
zablocki@informatik.uni-freiburg.de

Advanced Information Systems
Summerterm 2011
07.06.2011

3. Exercise Sheet: Transactions

Discussion: 21.06.2011

Submission Guidlines: We will discuss the solutions to the exercise sheet on 21.06.2011. If you want to have comments on your solutions you can submit them after the lesson.

Exercise 1 (Comparison of 2PL and TO)
Give examples which demonstrate:

a) There exist schedules which can be accepted by 2PL, but cannot be acccepted by TO. Assume first, that timestamps are assigned according to the order of the first actions of the transactions. Assume then, that timestamps can be defined in an arbitrary way. Are your examples still useful?
b) There exist schedules which can be accepted by TO, but cannot be accepted by 2PL.

Exercise 2 (Serializability)
Consider the following three schedules S_1, S_2 and S_3, where action C stands for COMMIT. Which of the schedules are

a) serializable,
b) snapshot-isolated,
c) multi-version serializable?

S_1:

T_0:	$Wxyz$											
T_1:		Rx	Wx				Ry	Wz	C			
T_2:				Rx	Wy					C		
T_∞:											$Rxyz$	

S_2:

T_0:	Wxy											
T_1:		Rx	Wx				Wy	C				
T_2:				Rx	Wy	C						
T_3:									Wx	C		
T_∞:											Rxy	

S_3:

T_0:	$Wxyz$													
T_1:		Rx				Wy	C							
T_2:			Rx		Wx			Wz				Ry	C	
T_3:				Ry					Wz	Rx	C			
T_∞:														$Rxyz$

Exercise 3 (Distributed 2PC)
In *centralized* 2PC we can distinguish three rounds of communication:

R_1: The coordinator contacts all participants,

R_2: each participant replies to the coordinator,

R_3: the coordinator instructs all participants.

In *decentralized* 2PC we can distinguish two rounds of communication:

D_1: The coordinator contacts all participants,

D_2: each participant replies to the coordinator and to all other participants.

a) Describe decentralized 2PC in more detail.

b) If there are $n > 2$ participants, how many messages are sent when performing centralized 2PC, respectively decentralized 2PC?

c) Assume a communication system with unlimited banwidth. Assume further, that the processor-time necessary to process a 2PC-algorithm is zero, both for the coordinator and the participants. If message transmission time ist t_0, how much time at least will it take when performing centralized 2PC, respectively decentralized 2PC?

Exercise 4 (Linear 2PC)
Linear 2PC protocol can be described as follows:

Assume all processes are linearly ordered: $P_1 \rightarrow P_2 \rightarrow \ldots\ldots P_n$ and the coordinator is P_1.

- The coordinator sends a message to its right neighbor (P_2) containing its vote, YES or NO. This message informs P_2 of the coordinator's vote and tells it to vote too.
- In general, a process P waits for a message from its left neighbor. If P receives YES and its own vote is NO or if it receives NO, then P forwards a NO to its right neighbor. If P receives YES and its own vote is YES, then P forwards YES to its right neighbor.
- If these rules are observed, then the rightmost process (P_n) will have all the information it needs to make a decision. If it receives a YES and its own vote is YES, then the decision is COMMIT, otherwise the decision is ABORT.
- Having made the decision, P_n sends a COMMIT or ABORT to its left neighbor informing it of the decision.
- Each process that receives the decision message decides accordingly and then forwards that message to its left neighbor. Eventually the message reaches the leftmost process (P_1) and the protocol ends.

In linear 2PC sending a message from a process to a neighbor is called a round.

a) If there are $n > 2$ participants, how many messages are sent when performing centralized 2PC, respectively linear 2PC?

b) Assume a communication system with unlimited bandwidth. Assume further, that the processor-time necessary to process a 2PC-algorithm is zero, both for the coordinator and the participants. If message transmission time is t_0, how much time at least will it take when performing centralized 2PC, respectively linear 2PC?