UNIVERSITÄT DES SAARLANDES
Fachrichtung 6.2 – Informatik
Prof. Dr.-Ing. G. Weikum
Dipl.-Inform. R. Schenkel

2. Übung zur Vorlesung *Transaktionsorientierte Informationssysteme* im Sommersemester 2000

2.1. For each of the following (input) schedules, show the output produced by 2PL, S2PL, SS2PL, TO, and SGT:
$s_1 = w_1(x)\ r_2(y)\ r_1(x)\ c_1\ r_2(x)\ w_2(y)\ c_2$
$s_4 = r_1(x)\ r_2(x)\ w_3(x)\ w_4(x)\ w_1(x)\ c_1\ w_2(x)\ c_2\ c_3\ c_4$

2.2. Under the 2PL protocol it is possible for transactions to "starve" in the following sense: A transaction gets involved in a deadlock, is chosen as the victim and aborted. After a restart, it gets again involved in a deadlock, is chosen as the victim and aborted, and so on. Provide a concrete example for such a situation, and describe how 2PL could be extended in order to avoid starvation of transactions.

2.3. Show that the wound-wait as well as the wait-die approaches to deadlock prevention both guarantee an acyclic WFG at any point in time.

2.4. Investigate the relationship between *Gen*(BTO) and *Gen*(2PL). Is one if the two more powerful than the other?

2.5. Reconsider the following condition for removing nodes (and edges) from the serialization graph in the SGT protocol. remove t_i when it is finished and none of the transactions that were active at the commit or abort of t_i are active anymore. Show that this condition, albeit seemingly natural, would lead to incorrect behaviour of the SGT protocol.

2.6. Consider the following input schedule of three concurrent transactions:

$$r_1(x)\ r_2(x)\ r_1(y)\ r_3(x)\ w_1(x)\ w_1(y)\ c_1\ r_2(y)\ r_3(z)\ w_3(z)\ c_3\ r_2(z)\ c_2$$

Which are the resulting output schedules under the BOCC and FOCC protocols? Remember that write steps are actually performed on private workspaces, the commit requests initiate the validation, and the write steps are performed on the shared database only after a successful validation.

2.7. Consider an alternative variant of BOCC where a transaction validates by comparing its write set against the read sets of concurrent, previously validated transactions. Would such a variant work correctly, i.e., ensure that all output schedules are CSR?

2.8. Construct an example execution that demonstrates that FOCC may provide non-serializable schedules if the critical-section condition for the val-write phases were dropped.

2.9. Develop variants of the BOCC and FOCC protocols that do no longer need the critical section for the val-write phases of transactions. Discuss the pros and cons of such a relaxation, in comparison to locking protocols.