Exercises for Transaction Systems, summer term 2017
Linnea Passing (linnea.passing@tum.de)
http://www-db.in.tum.de/teaching/ss17/transactions/

Sheet No. 8

Info

- Due date: Friday, July 14, 3pm.
- Please send your solution via e-mail, and prefix the subject with [transactions].
- Please include your Matrikelnummer and your name.

Exercise 1 (5 points) Prove: In the “action” model, where each step is a combination of a read operation immediately followed by a write operation on the same data, MVSR = VSR.

Hints:

0. MVSR ⊇ VSR in the “normal” model. Thus you have to show that the action model restricts the MVSR class. Let m be the multi-version schedule, and m′ be a view-equivalent serial mono-version schedule.

1. Think about whether the order of two writes to the same data item can change when transforming m to m′. What about two reads? Show that the potential for “moving operations around” is more limited in the action model than in the normal model.

2. Show that these restrictions force write-read pairs (these pairs constitute the reads-from relation that you need for VSR) to not change their order when transforming m to m′.

3. Show that in m′ these write-reads pairs are monoversion (i.e. the read always reads the latest written version)

4. Now you should be able to show “When I can transform m to an MVSR-equivalent m′, m′ is automatically also in VSR”.

Exercise 2 (10 points) Consider a database with a person table (unique name, city). Two operations:

- select(c): select * from person where city = c;
- update(n,c): update person set city = c where name = n;

A B+-tree for both attributes exists. It has height 2 (i.e., root and leaves). The operations are: lookup (search(key)), record fetch (fetch(rid)), record modification (modify(rid)), index maintenance (insert(key,rid) and delete(key,rid)). All operations are transformed into page reads and writes.

We consider two transactions:

- T_1 finds all persons from Munich and Garching.
- T_2 moves a couple (John and Jane Doe) from Munich to Garching.

Model them as 3-level transactions. Give a non-serial example for a 3-level schedule that (a) is tree-reducible
(b) is not tree-reducible

**Exercise 3 (7 points)** In a semi-queue, enqueuing works like in a usual queue, but dequeuing non-deterministically selects and removes an arbitrary entry from the queue. Construct the return value commutativity table for a semi-queue. Show (by example) that semi-queues allow higher concurrency than usual queues.