Database Systems WS 2023/24
Exercise 11: Distributed 22.01.2024, Due 29.01.2024 12:00 MEZ
Submitted by Erik Schwede, Suma Keerthi

1 Schedules - RC, ACA, Strict (1 P.)

For the following schedules, decide in which of the classes RC, ACA, or ST they are.

$$
\begin{aligned}
s_1 &:= r_1(a)\ w_1(a)\ r_2(a)\ r_2(b)\ r_2(a)\ r_1(b)\ c_1\ w_2(b)\ c_2 \\
s_2 &:= w_1(b)\ r_1(a)\ r_2(a)\ r_1(b)\ w_2(a)\ w_1(b)\ w_2(b)\ c_2\ c_1 \\
s_3 &:= r_2(a)\ r_1(b)\ r_1(a)\ w_2(a)\ w_1(b)\ w_3(b)\ r_3(b)\ c_3\ c_1\ r_2(b)\ c_2
\end{aligned}
$$

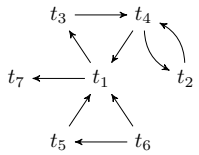
2 2-Phase-Locking and Waits-for-Graph (1 P.)

1. Given the following schedules:

$$
\begin{aligned}
s_1 &= r_1(c)\ w_2(a)\ r_1(a)\ r_3(b)\ w_3(a)\ c_3\ w_2(c)\ w_1(c)\ c_1\ c_2 \\
s_2 &= w_1(b)\ r_2(c)\ w_3(a)\ w_2(b)\ r_1(a)\ c_1\ c_3\ w_2(a)\ c_2
\end{aligned}
$$

   Create a 2PL history for both schedules and note the Waits-for-Graph (WfG) every time it changes. If you encounter a deadlock, abort the transaction that caused the deadlock (No deadlock prevention strategy).

2. Given the following WfG. Which nodes are chosen for the strategies **most cycles** and **most edges**.



3 Deadlock prevention (1 P.)

Implement a scheduler, which creates a (SS)2PL schedule, using the deadlock prevention strategies wait-die and immediate restart.

Test your program with the given schedule. It has to print the output schedule for each of the prevention strategies and indicate when a transaction has to wait or abort. After a transaction waits or aborts, you do not have to continue/restart it in the solution history.

$$s := w_1(x)\ r_2(x)\ w_3(y)\ r_1(y)\ r_3(z)\ w_1(x)\ c_1\ w_2(y)\ c_2\ w_3(y)\ c_3$$

You can use the template provided in OLAT, which implements an internal representation of a history and all required operations.

Submit the code as separate file and submit the output of running your program with the solution PDF.

4 Timestamp-Based Approaches (1 P.)

$$s_1 = w_1(x)\ r_2(x)\ w_2(x)\ r_1(x)$$
$$s_2 = r_1(y)\ w_1(x)\ r_2(x)\ w_3(x)\ w_1(y)\ w_2(x)\ r_1(x)$$

Describe how a timestamp ordering (TO) based scheduler would execute the operations. Complete the following tables. Note the used rule in the "Comment" column. The max-r and max-w columns contain the value of max-q-scheduled for the given object. One row contains the entry after applying the rule. You may assume $ts(t_1) < ts(t_2)$. Do not restart aborted transactions.

s_1:

Operation	max-r(x)	max-w(x)	Comment
BOT_1	$-\infty$	$-\infty$	$ts(t_1) = 1$
BOT_2	$-\infty$	$-\infty$	$ts(t_2) = 2$

s_2 without Thomas' write rule:

Operation	max-r(x)	max-w(x)	max-r(y)	max-w(y)	Comment
BOT_1	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$ts(t_1) = 1$
BOT_2	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$ts(t_2) = 2$
BOT_3	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$ts(t_3) = 3$

s_2 with Thomas' write rule:

Operation	max-r(x)	max-w(x)	max-r(y)	max-w(y)	Comment
BOT_1	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$ts(t_1) = 1$
BOT_2	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$ts(t_2) = 2$
BOT_3	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$ts(t_3) = 3$