Exercises
Distributed Systemes: Part 2
Summer Term 2013
5.7.2013

2. Exercise sheet: Distributed Concurrency Control and Reliability

Exercise 1
Give an example of a serializable schedule that has been generated by a timestamp-based scheduler that could not have been generated by a 2PL scheduler.

Exercise 2
Consider the distributed waiting graph for transactions $T_1$ to $T_6$ that are executed at sites 1, 2, and 3 (cf. Figure 1). Assume that $T_1$ is requesting a lock at site 1 which is already occupied by $T_2$. Simulate the path pushing algorithm to detect a deadlock and give the resulting messages that are exchanged.

![Figure 1: Distributed waiting graph](image)

Exercise 3
a) Find a global schedule using the procedure with explicit tickets for heterogeneous federations. Consider the local transaction $T_1$ and the global transactions $T_2$ and $T_3$ (for 2 sites with $D_1 = \{C, D\}$ and $D_2 = \{A, B\}$).

- $T_1 = RC\ WC\ RD\ WD$
- $T_2 = RC\ WC\ RA\ WA$
- $T_3 = RD\ WD\ RB\ WB$

Extend schedule $S$ with the corresponding take-a-ticket-operations. Does an equivalent serial global scheduler exist for schedule $S$? (Hint: Analyse the local/global conflict graphs).
\[ S = \begin{align*}
\text{Site 1:} & \quad R_1 C W_1 C \quad R_1 D W_1 D \quad R_2 C W_2 C \quad R_3 D W_3 D \\
\text{Site 2:} & \quad R_2 A W_2 A \quad R_3 B W_3 B
\end{align*} \]

b) Prove the following statement: All schedules that the procedure with explicit tickets accepts are serializeable.

**Exercise 4**

Consider this notation for ordered messages that occur for a 2P-commit: \((i, j, M)\) means that site \(i\) is sending a message \(M\) to site \(j\), where the value of \(M\) is either P (prepare), R (ready), D (don’t commit), C (commit), or A (abort). Assume that site 0 is the coordinator and sites 1 and 2 the participants. The following example shows a possible ordering of messages that result in a successful transaction:

\((0, 1, P), (0, 2, P), (2, 0, R), (1, 0, R), (0, 2, C), (0, 1, C)\)

a) Give an ordering of messages that describes the following situation: site 1 requests a commit, site 2 requests an abort.

b) How many possible orderings are there for a successful transaction?

c) If site 1 wants to commit, but site 2 does not, how many orderings of messages are there for this situation?

d) If site 1 wants to commit, site 2 is not reachable and does not answer, how many orderings of messages are there for this situation?