STiDC'08: Exercise 5

January 12, 2009

1 Problem 1

Prove that it is impossible to implement a consensus object using only queues and atomic registers in a system of 3 processes.

Solution. A solution can be found in [Herlihy, M. P. Wait-free synchronization. ACM Transactions on Programming Languages and Systems, 13(1):124—149, January 1991] (the first reference on the course web site), Theorem 7, page 13 (136).

2 Problem 2

Devise an algorithm that implements a consensus object using (any number of) queues that are *initially empty* and atomic registers in a system of 2 processes.

Solution. The main idea is to use the algorithm that implements a consensus in a system of 2 processes using initialized queues (Algorithm 1) and to add a new "initialization" phase.

procedure $cons_i(Q, R, val_i)$
- $R[i] \leftarrow val_i$
- $q_i \leftarrow Q.deq()$
- **if** $q_i = \text{"winner"}$ **then return** val_i
- **else return** $R[3-i]$

Algorithm 1: Consensus in a system of 2 processes using initialized queues.

Algorithm 1 assumes that Q is initialized to $\langle \text{"winner"}, \text{"loser"} \rangle$.

procedure $propose_i(val_i)$
- $Q_i.enq(\text{"winner"})$
- $Q_i.enq(\text{"loser"})$
- $ready_i \leftarrow true$
- **for** $k \leftarrow 1, 2$ **do**
  - **if** $ready_k$ **then** $val_i \leftarrow cons_i(Q_k, R_k, val_i)$
- **return** val_i

Algorithm 2: Consensus in a system of 2 processes using non-initialized queues.

In Algorithm 2 both processes first initialize their queues and then they execute the Algorithm 1 on all queues that have been initialized using the last decided value. Algorithm 2 uses queues $Q_{1,2}$ (initially empty), registers $R_{1,2}[1,2]$ and registers $ready_{1,2}$ (initalized to *false*) as depicted in Figure 1.

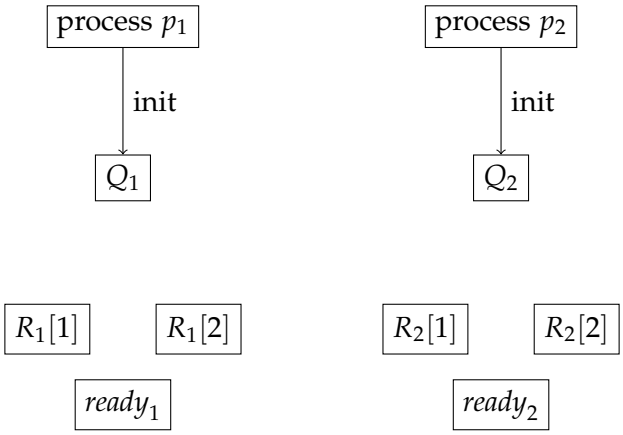

Figure 1: Objects used in Algorithm 2.