Concurrent Algorithms

Exercise 10 Solutions

Let a correct process be a process that does not crash. Then obstruction-freedom stipulates the following:

- An implementation (of a shared object) is obstruction-free if any of its operations returns a response if it is eventually executed without concurrency by a correct process.

Wait-freedom is stronger: any correct process that executes an operation eventually returns a response. The difference is concurrency. Obstruction-freedom ensures termination in an obstruction-free execution, i.e., assuming that eventually at most one process is taking steps. However, in other executions, an obstruction-free implementation can never terminate.

The implementation is obstruction-free. Suppose that eventually only process P is taking steps. Then eventually P finds its local timestamp ts is the highest among all the values in the registers in array T, and then returns a value.

Now we give an example execution where the implementation violates agreement, which shows the implementation is incorrect. Figure 1 illustrates the example execution. Assume two processes P_1 and P_2.

1. P_1 proposes some value v_1. P_1 executes until the condition $ts = maxts$. P_1 checks the condition to be true. Then P_1 is suspended.

2. P_2 proposes some value v_2. P_2 executes to the end. We note that in the first loop, P_2 sees that each cell of an array V is $(\perp, 0)$ and thus P_2 assigns v_2 to val after the first loop. Then P_2 decides v_2.

3. P_1 now continues and decides v_1.

The example execution breaks agreement as P_1 and P_2 returns their own proposals, which can be different.

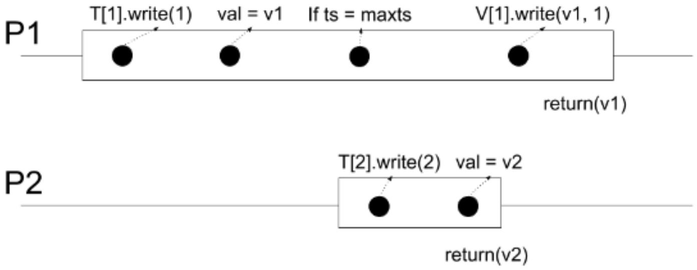

Figure 1: Example execution of an incorrect implementation of obstruction-free consensus