Concurrent Algorithms November 18, 2019

Solution to Exercise 7

Algorithm 1 Obstruction-free consensus

```
1: ▷ Shared variables
2: CA[0, . . . , ∞]  ▷ infinite array of commit-adopt objects in their initial state
3:
4: ▷ Process p_i proposes value v
5: procedure PROPOSE(v)
6:     j ← 0
7:     val ← v
8:     while true do
9:         res ← CA[j].propose(val)
10:        if res = commit(v') then return v'
11:        else if res = adopt(v') then
12:            j ← j + 1
13:            val ← v'
```

On a high-level, Algorithm 1 operates as follows. Initially each process proposes its value v stored in *val* (Line 7) in the first commit-adopt object ($CA[0]$). If the commit-adopt object returns *commit*, then the algorithm terminates, otherwise the algorithm uses the next commit-adopt object where it proposes the value v' it received from $adopt(v')$ (Line 11). The process keeps proposing values to subsequent commit-adopt objects as long as it receives a *adopt* response.

Algorithm 1 implements obstruction-free consensus (i.e., obstruction-free termination, validity, and agreement):

- **Obstruction-free termination** follows from the progress and commitment properties of the commit-adopt objects. If some process p eventually executes alone, then it eventually reaches an index i in the CA array such that it is the only process to invoke *propose* on $CA[i]$. By the commitment and progress properties of $CA[i]$, p must receive $commit(v')$ (for some value v') from $CA[i]$ at line 9. Thus, p will decide v' at line 10.
- **Validity** follows immediately from the validity property of the commit-adopt objects.
- **Agreement**. Assume by way of contradiction that Algorithm 1 does not satisfy agreement. This means that there are two processes p_a and p_b such that p_a decides v_a and p_b decides v_b where $v_a \neq v_b$. Process p_a received a *commit* response from commit-adopt object with index ca_a and process p_b received a *commit* response from commit-adopt object with index ca_b. Naturally $ca_a \neq ca_b$ since otherwise $v_a = v_b$ (due to the agreement property of commit-adopt), a contradiction. Assume without loss of generality that $ca_a < ca_b$. This means that when process p_b proposed to object $CA[ca_a]$ it received $adopt(v_a)$, hence process p_b subsequently proposed v_a to

$CA[ca_a + 1]$. This is the case for all other processes as well: all processes receive v_a when proposing to object $CA[ca_a]$, hence all processes propose v_a to $CA[ca_a + 1]$. Due to the commitment property of the commit-adopt object, all subsequent commit-adopt objects $CA[k]$ with $k \geq ca_a$ commit value v_a. Hence $CA[ca_b]$ also commits v_a, a contradiction.