York University CSE 6117 March 5, 2016

# Homework Assignment #6
# Due: March 14, 2016 at 5:30 p.m.

**1.** Consider an asynchronous shared-memory system with $n$ processes and up to $f$ halting failures. An algorithm may specify the initial state of all shared objects. In the consensus problem, each process receives an input value (a natural number) and the processes' outputs must satisfy the following conditions.

- Termination: in every execution where at most $f$ processes experience halting failures, all non-faulty processes eventually output a value.
- Agreement: in every execution, all outputs produced in the execution are identical.
- Validity: in every execution, each output is the input of some process in that execution.

**(a)** Give a simple consensus algorithm that uses one shared stack, together with shared read-write registers for the case where $f = 1$. (Your algorithm should work for every $n$.)

**(b)** Now you will prove that consensus is not solvable using stacks and read-write registers if $n = 4$ and $f = 2$, even if the inputs are restricted to be from the set $\{0, 1\}$. We modify some of the definitions from class as follows.

A configuration $C$ of an algorithm is called *$v$-valent* (for $v \in \{0, 1\}$) if in every execution that passes through configuration $C$ and in which at most two processes experience halting failures, all live processes eventually output $v$.

A configuration is *univalent* if it is 0-valent or 1-valent. Otherwise, it is *multivalent*.

To derive a contradiction, suppose there is an algorithm $A$ that solves consensus for $n = 4$ and $f = 2$ using stacks and read-write registers.

(i) Show that $A$ has a multivalent initial configuration.

(ii) Let $C$ be any multivalent configuration of $A$ and let $P$ and $Q$ be any two processes. Prove that if $C$ is a multivalent configuration of $A$, then at least one of the following two configurations is also multivalent:

- $C_{PQ}$, the configuration obtained from $C$ by allowing $P$ to take one more step and then $Q$ to take one more step.
- $C_{QP}$, the configuration obtained from $C$ by allowing $Q$ to take one more step and then $P$ to take one more step.

Hint: the arguments used in class require some modifications, because you can no longer assume that if a process runs by itself it will eventually generate an output; termination only guarantees that a process will output a value if at least two processes continue to take steps.

(iii) Show that there exists an infinite execution of $A$ in which two processes take infinitely many steps without ever outputting a value. (This contradicts the termination property.)