Distributed Algorithms 236755
Spring 2011, Home Assignment 3

1. A solo-terminating with fail consensus object $C$ is invoked by $C.propose(x)$, which returns an answer $y$ that can be either a value or fail, such that:

   - **Agreement:** If an invocation of $C.propose(x)$ returns $y = v$ which is not fail then any other invocation of $C.propose(x')$ returns $y = v$ or $y = fail$.
   - **Validity:** If an invocation of $C.propose(x)$ returns $y = v$ which is not fail then there was an invocation of $C.propose(v)$ by some process, which does not return fail.
   - **Solo-Termination with fail:** If a process runs alone during a whole invocation of $C.propose(x)$ then it returns $y \neq fail$. Any other invocation of $C.propose(x)$ eventually returns a value $y$ which is either fail or some value $v$.

   Show that there is no implementation of such an object using read/write registers.

   Guidelines: show this impossibility by a reduction to wait-free consensus for two processes, i.e., show that given such an implementation of a consensus object $C$, there is a wait-free consensus algorithm for two processes (which we proved that does not exist).

   Hints: the algorithm for $p_0$ and $p_1$ does not have to be symmetric. Also, you may use additional read/write registers.

2. This question considers the lower bound proof for synchronous consensus.

   (a) Prove by induction the Connectivity Lemma from the tutorial about uniform consensus in a message-passing system:

   If $X = L^0(X)$ is similarity connected where in every configuration no process is faulty, then for every $k \leq t$, the set $L^k(X)$ is similarity connected where in every configuration there are at most $k$ faulty processes.

   Hint: in the induction step, first prove that for every $C \in L^{k-1}(X)$, the set $L(C)$ is similarity connected, and then prove that $L^k(X) = \bigcup_{C \in L^{k-1}(X)} L(C)$ is similarity connected.

   (b) Use the Connectivity Lemma to prove a $f + 1$ lower bound for the number of rounds in a synchronous consensus algorithm. Assume that $t < n - 1$.

   Hint: show that there is a configuration in $L^k(X)$ in which some process does not decide.
Algorithm 1 $k$-set consensus algorithm: code for process $p_i$

1: CONSENSUS($x_i$) 
   
   local variable $V$, initially $V = \{x_i\}$ 

2: for all $r$, $1 \leq r \leq \frac{f}{k} + 1$ do 

3: send $V$ to all processes 

4: receive $S_j$ from $p_j$, $0 \leq j \leq n - 1$, $j \neq i$ 

5: $V := V \cup \bigcup_{j=0}^{n-1} S_j$ 

6: if $r = \frac{f}{k} + 1$ then $y := \min(V)$ 

7: end if 

8: end for 

9: end

3. Design an early-stopping consensus algorithm for crash failures: If $f$ processes fail in an execution, then the algorithm terminates within $O(f)$ rounds. Prove that the algorithm satisfies this property.

   Hint: processes need not decide in the same round.

4. Define the $k$-set consensus as follows. Each processor starts with some arbitrary integer value $x_i$ and should output an integer $y_i$ such that

   - **Validity**: $y_i \in \{x_0, \ldots, x_{n-1}\}$ processes.
   - **$k$-Agreement**: The number of different output values is at most $k$.

Show that Algorithm 1 solves the $k$-set consensus problem in the presence of $t$ crash failures, for any $t < n$.

What is the message complexity of the algorithm?

Submission date: 29/5/2011, 16:00, in pairs. Please submit to Eshcar’s mailbox on the fifth floor.