Exercise 3: Impossible!

Task 1: Stop Failing, You Cowards!

The goal of this exercise is to show that under the synchronous message passing model, for any consensus algorithm there are executions with f crashes in which solving consensus requires at least $f+1$ rounds. As we want to prove a lower bound, we assume a fully connected communication graph.

Here are some helpful definitions, as the tools "b-valent, bivalent, fair" are not defined for the synchronous case: We will assume that a crashing node still attempts to send messages, and the adversary chooses the subset of delivered messages.

Let $\mathcal{E}_0, \mathcal{E}_1$ be a pair of partial executions which are indistinguishable for all nodes except v, and whose maximal fault-free extensions have different outputs. Then we call this node v to be *pivotal*, as only this node's state makes a difference.

Note that maximal fault-free extensions are unique.

a) Show that there is a pair of inputs (round 0) with a pivotal node, which we will denote v_0.

   Hint: Use the same argument as for the asynchronous case.

b) Prove that, given a pair of r-round executions (with $r \leq n-3$) with a pivotal node v_r, crashing the node "in the right way"[1] yields a pair of $(r+1)$-round executions with a new pivotal node v_{r+1}.

   Hint: The reasoning is similar as for a), but the "inputs" are replaced by the messages of v_r in round r of each of the executions — or their absence due to the node crashing.

c) Conclude that for any $f \leq n-2$, there are executions with at most f faults in which some node neither crashes nor terminates earlier than round $f+1$.

d)* For a small but fixed $n > 1$ (e.g. 2 or 3), find a fault-tolerant algorithm that solves consensus for an arbitrary number of faults, and for $f = n-1$ takes only f rounds. Conclude that the result of c) is tight. This is to show that not only is $f = n$ a special case, but $f = n-1$ is a *different* special case, too!

[1] this includes not crashing the node at all

Task 2: Impossible? We'll Do it in $f + 2$ Rounds!

The goal: matching the lower bound with an upper bound.
The connectivity: complete.
The model: synchronous message passing.
The task: consensus.
The challenge: crash faults.

a) Suppose each node maintains a bit p_i. In each round, each node sends its bit to all other nodes and sets it to 0 if it received a 0.[2] Show that if a node receives messages from the same set of senders in two consecutive rounds and either all are opinion 0 or all are opinion 1, all nodes have the same bit p_i.

b) Use this observation to construct a synchronous consensus algorithm tolerating an arbitrary number of faults.

c) Prove that the algorithm is correct and terminates in at most $f + 3$ rounds in executions with at most f faults (if necessary, modify your algorithm to achieve this property).

d)* Modify the algorithm to terminate in $f + 2$ rounds!

Hint: In contrast to the $f + 3$, nodes will need to use their knowledge of n. This subtask is not as easy as it seems!

Remark: Note that the algorithm can deal with an arbitrary number of faults, yet the running time is bounded in terms of the *actual* faults happening. This property is called *early-stopping*. As faults are supposed to be uncommon events, that's pretty neat!

Task 3*: Intense Sharing

a) Find out what the term "consensus number" refers to!

b) Ponder the consensus number of shared memory that, besides atomic reads, permits to write to up to $k > 1$ shared registers in a single atomic step!

c) Share your insights in the exercise session!

[2] Not vice versa. This is one-sided. A node never changes its opinion to 1.