Exercise 11: Counting

1

The goal of this exercise is to understand the consistency properties of the bounded max register implementation from the lecture.

a) Show that if one always writes to $R_<$ if $i < M$, regardless of whether switch reads 0, the implementation is not linearizable! (Hint: Start a read operation that reads 0 from switch, complete a write operation for $i \geq M$, then another one for $0 < i < M$. Show that the order implied by the "precedes" relation now is incompatible with any sequential execution of the max register!)

b) Show that if a write operation (for $i < M$) reads switch $= 1$, there is a preceding write operation for $i \geq M$. Conclude that it is always possible to determine a valid linearization point for such an operation.

c) Prove that the max register of maximum value $2M$ constructed from two max registers of maximum value M and a read/write register is linearizable. (Hint: Divide operations into three classes: (i) writes of $i < M$ and reads reading switch $= 0$, (ii) write operations for $i < M$ reading switch $= 1$, and (iii) writes for $i \geq M$ and reads reading switch $= 1$. Order operations from classes (i) and (iii) first and then apply b) to handle those in class (ii).)

2

In this exercise,[1] we're going to implement more powerful registers from weak ones. We start with very simple registers. They are

- *binary*, i.e., can hold only values 0 and 1,
- *single-writer*, i.e., only one node may write them,
- *single-reader*, i.e., only one node may read them, and
- *safe*, i.e., they guarantee that (i) *some* legit value is returned, but (ii) only if the most recent write operation is complete,[2] it is certain that it is the written value.

All registers are initialized to 0 in this exercise.

a) Implement a *regular* binary single-writer single-reader register from a safe one. A regular register is a safe register that guarantees that only values of overlapping or the latest preceding write are returned (or the initial value, if there is no preceding write). (Hint: Do not actually write until unless the content of the register is changed.)

b) Implement a regular M*-valued* single-writer single-reader register from M regular binary single-writer single-reader registers. An M-valued register can take values $0, \ldots, M-1$. (Hint: Use the i^{th} register to represent value $i-1$. Read in ascending order, but write in descending order.)

[1] Don't count the number of parts of this exercise – or at least don't use it as complexity measure. All parts but e) are very straightforward (one page total in the sample solution); e) is not too difficult either, but not as compressible in terms of write-up. Given that you're wading through decades of research in a single exercise, it's still very compact!

[2] Note that because there is only a single writer, we can require that there is never more than one write in progress.

c) Implement a *linearizable* M-valued single-writer single-reader register that can be written $W-1$ times from a regular MW-valued single-writer single-reader register. (Hint: Use timestamps, and let the reader always return the value for the largest timestamp.)

d) An *n-reader* register is one that can be read by n different nodes. Show that naively using n atomic single-writer single-reader registers to construct a single-writer n-reader register does *not* result in a linearizable implementation.

e) Construct a linearizable M-valued single-writer n-reader register that can be written $W-1$ times out of n^2+n atomic MW-valued single-writer single-reader registers. (Hint: Use timestamps and leverage the additional n^2 registers to communicate between the readers. The readers will read from "their" incoming registers, then from the writer's register, then write the timestamp/value pair of the maximum seen timestamp to their outgoing registers, and only then return the respective value.)

f) Construct a linearizable M-valued n-writer n-reader register that can be written $W-1$ times out of n atomic MW-valued single-writer n-reader registers. (Hint: Let writers read all registers first and write with a timestamp larger than all timestamps they read.)

g) Conclude that for any bounded number of operations, safe binary single-writer single-reader registers are as computationally powerful as atomic multi-valued multi-writer multi-reader registers. (Hint: Concentrate on *not* thinking about efficiency. DO NOT THINK ABOUT EFFICIENCY!)

3*

Consider a fully connected asynchronous message passing system.

a) Implement a wait-free linearizable single-writer single-reader register!

b) It turns out that this didn't work. Why?

c) Check out what sort of simulations are around in the literature.

d) Write what you've learned to the green shared memory in the exercise session for everyone else to read!