Distributed Algorithms CPSC-561
Assignment 2

Michael McCulloch

1 LL/SC

For all of the following algorithms, they are each wait-free, as they do not contain a loop.

A) CAS from SCAS

Algorithm 1 CAS from SCAS

function CAS_p(old,new)
    old' = SCAS.scas(old, new)
    return old' = old
end function

Let the invocation of SCAS.scas() be the linearization point X. *Claim:* If a process P executes X before another process Q executes X, then the invocation of P is ordered before the invocation of Q. *Proof:* By the definition of SCAS, if it returns the same value as it's first argument, it must have succeeded. If this value is different, another process Q must have finished it's invocation between the invocation of SCAS and the assignment of it's result to old' within P, a contradiction.

B) SCAS from CAS

Algorithm 2 SCAS from CAS

```
function SCAS_p(old,new)
    success = CAS.cas(old, new)
    if success then
        return old
    else
        return CAS.read()
    end if
end function
```

Let the invocation of CAS.cas() be the linearization point X. *Claim: see above*. *Proof:* If CAS is successful, then the old value passed in must be the old value. If CAS is unsuccessful, then a different value must have been the old value, which can only have been cause if another process executed X first. A contradiction.

C) CAS from LL/SC

Algorithm 3 CAS from LL/SC

```
1: function CAS_p(old, new)
2:     old' = LL/SC.LL()
3:     if old' ≠ old then
4:         return false
5:     else
6:         return LL/SC.SC(new)
7:     end if
8: end function
```

Claim: If a process P executes lines two through six, before another process Q executes the same, then P's invocation is ordered before Q's. *Proof:* If P returns `false`, then another process must have executed either LL() or SC() successfully ...FILL IN THE BLANK...?

2 $\mathbb{Z}_k$-Counter

Claim: There is no consensus-3 solution using $\mathbb{Z}_k$-counters. Proof by cases for 3 processes: A, B and C.

2.1 Case: Different $\mathbb{Z}_k$-counters

w.l.o.g. assume A and B increment different $\mathbb{Z}_k$-counters. Let K be a bivalent configuration, π_A and π_b be the steps that A and B take, respectively, to increment their own counters ...FILL IN THE BLANK... then $\pi_A(K)$ and $\pi_A\pi_B(K)$ are indistinguishable to A

2.2 Case: Different/Same Registers

See proof in lecture notes.

2.3 Case: Same $\mathbb{Z}_k$-counters

A and B increment A $\mathbb{Z}_k$-counters. Let K be a bivalent configuration, π_A and π_b be the steps that A and B take, respectively, to increment the counter. Assume for contradiction that π_R causes the configuration to be R-critical. If the counter's initial value is c, then $\pi_A(K)$ causes it to be $c+1 \bmod k$. the same is true for $\pi_B(K)$. Hence, after $\pi_A\pi_B(K)$, the value of the counter is $c+2 \bmod k$ and the same is true for $\pi_B\pi_A(K)$. Therefor, to a third process C, these two states are identical, and therefor are not critical, yielding a contradiction.

3 SRSW