York University | EECS 6117 | February 3, 2023

Homework Assignment #3
Due: February 9, 2023 at 5:00 p.m.

In both questions of this assignment, we consider a synchronous message-passing system with no failures where processes have unique identifiers drawn from the set $\{1, \ldots, N\}$ and all processes know n, the number of processes.

1. In class we saw an algorithm for a ring network that takes $O(\log^* N)$ rounds to colour the nodes of the network with three colours. We assumed the ring was oriented: each process knows which of its two neighbours is its clockwise neighbour and which is its counter-clockwise neighbour. The algorithm from class is described below. Assume the local variable *colour* is represented in binary.

```
colour ← binary representation of my identifier
for 2 log* N rounds do
    send colour to counterclockwise neighbour
    receive neighbourColour from clockwise neighbour
    diff ← rightmost position where binary representations of colour and neighbourColour differ
    bit ← value of bit in position diff of colour
    colour ← diff · bit          ▷ concatenate binary representation of diff with bit
end for
▷ At this point, all colours are from {1, 2, 3, 4, 5, 6}
for 3 rounds do
    send colour to both neighbours
    receive colourLeft and colourRight from my two neighbours
    if colour > colourLeft and colour > colourRight then
        colour ← min({1, 2, 3} − {colourLeft, colourRight})
    end if
end for
```

Consider an *unoriented* ring where each process initially knows it has two neighbours but does not know which is its clockwise neighbour and which is its counter-clockwise neighbour. (In other words, each node has neighbour #1 and neighbour #2, but there is no consistent numbering of neighbours: it could be the case that in some nodes, neighbour #1 is the clockwise neighbour, while in other nodes, neighbour #1 is the node's counter-clockwise neighbour.)

Adapt the way that colours are refined in the first loop so that the algorithm works in an unoriented ring. Your (deterministic) algorithm should still run in $O(\log^* N)$ rounds. Briefly justify why each round maintains the invariant that each node's current colour is different from its neighbours.

2. An independent set in a graph is a set S of nodes such that no two nodes in S are connected by an edge. We wish to design a distributed algorithm to compute an independent set in an oriented ring: each process in the set should output 1 and each process outside the set should output 0.

(a) Give a deterministic algorithm to compute an independent set in an oriented ring such that the number of nodes in the independent set is at least $\frac{n}{10}$. Your algorithm should run in $O(\log^* N)$ rounds

Hint: start by colouring the ring with 3 colours.

(b) Give a randomized distributed algorithm to compute an independent set in an oriented ring in $O(1)$ rounds such that the *expected* number of nodes in the independent set is at least $\frac{n}{10}$.