ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

FS 2015

Prof. R. Wattenhofer
Pascal Bissig

Principles of Distributed Computing
Sample Solution to Exercise 2

1 Leader Election in an "Almost Anonymous" Ring

a) Yes, it is possible:

Algorithm 1 Leader Election (all but one nodes have the same ID)

```
1: send ID to each neighbor
2: if both received ID's differ from the own ID then
3:     I am the leader
4: end if
```

Note that this only works for $n \geq 3$. For $n = 2$ both nodes receive only one ID and the node with the lower ID can become the leader. For $n = 1$ the node does not receive a message and thus becomes the leader.

b) No, not always: If both differing nodes have the same ID, the number of nodes is even, and these nodes are exactly opposite to each other, their local views will always remain completely identical. Thus they both must take the same decision or never terminate.[1]

[1] In an odd, directed ring, however, symmetry can be broken by means of the distance between the two nodes.

2 Distributed Computation of the AND

a) Assume for contradiction that an algorithm A exists solving the problem. Examine what happens on a ring of $n \geq 3$ nodes, where all inputs are 1. In any round, the states of all nodes are identical (by induction, as in the proof of Lemma 2.4). Observe that these states do not depend on the size n of the ring, as the algorithm is uniform and the local topology is independent of n. Thus, there must be some constant round number t that does not depend on n such that all nodes terminate and output 1 in round t (as we assumed the algorithm to be correct).

Now run A on a ring of size $2(t+1)$, where exactly one node has 0 as input bit. Up to distance t from the node on the opposite side of the ring, all nodes have input 1. Again, analogously to Lemma 2.4, until round t, this node will have the same state as if in a ring where all nodes have input 1. Hence, in round t it will terminate and output 1, contradicting the assumption that A is correct and should output 0 at all nodes.

b) All input values have to be sent all around the ring. In order to detect the returning of the own message, we add a hop counter to each message. If the message has made n hops, it has arrived where it started.

c) The following algorithm calculates the AND in a synchronous, non-uniform ring:

Algorithm 2 AND in the Ring: synchronous, non-uniform (n is the number of nodes)

```
1: if input bit = 0 then
2:    send 0 to the neighbor in the ring
3: end if;
4: for i := 2 to n do
5:    if received a 0 and have not already sent a 0 then
6:       forward the 0 to the other neighbor in the ring
7:    end if
8: end for;
9: if received at least one 0 then
10:   result := 0
11: else
12:   result := 1
13: end if;
```

If the result is 1, no message is sent, otherwise there is exactly one message over each link. Thus, time and message complexity are both n.