1 Leader election in synchronised undirected ring

The goal is to elect a leader in an undirected ring with n nodes, that communicate in synchronous rounds. We assume that the processors are identical, except for unique identifiers. The size of the network is unknown. Each node has a list of its neighbours in an arbitrary order.

The most straightforward solution is to make every node send its identifier to both neighbours. By passing the identifiers around processors can choose the node with the highest identifier to be the leader. This algorithm requires only $O(n)$ rounds, but it needs $O(n^2)$ messages to be sent. Here we present a synchronous version [2] of an algorithm presented in 1980 by Hirschberg and Sinclair [1], which only requires $O(n \log n)$ messages.

2 The algorithm

The idea is that nodes initiate messages that are passed around in both directions along paths of predetermined length (which are successive powers of 2). In the first round every processor sends a message containing its identifier, length 1 and an information that the message is going out. Then in every round they check for messages from both neighbours. There are five main cases:

1. If the identifier in the message is higher than the processor's identifier:
   (a) If the message already visited the predetermined number of nodes, then we send it back, with the information that it's coming back.
   (b) If the message have not yet visited the predetermined number of nodes or is already coming back, then we pass it to the next processor lowering its length by 1.
2. If the identifier in the message is lower than the processor's identifier, then the message is ignored.
3. If the identifier in the message is equal to the processor's identifier and the message is going out, then we win the election and now need to send an ending message to both neighbours.
4. If the identifier in both messages is equal to the processor's identifier and the messages are coming back, then we initiate new message with double the length of the previously initiated message and send it to both neighbours.
5. If the message is an end message, we pass it to the next processor and ignore the other one.

3 Correctness

Let v be the node with highest identifier. Any node that receives a message from v must have a lower identifier. That means it will have to pass this message further or send it back to the source. It follows that every message with length lower than n will come back to v, so after $\lceil \log n \rceil$ initiated messages v will send one with a length at least n. This message will visit v again without being sent back at any point, which means v will be declared the leader.

On the other hand for any node $w \neq v$ it is impossible to be declared the leader, because there exists $k < n$ such that a message with length k sent from w will visit v, where it will be

ignored. Every such node will be relaying messages until it receives a stop message originating from the leader.

4 Complexity analysis

Theorem 1. *Number of rounds is $O(n)$.*

Proof. If a message with length l is not ignored at some point, it must come back exactly in $2l$ rounds. Every node starts with a message with length 1 and every time the message comes back, a new one with double the length is initiated. We know that a leader is elected after receiving a message with length $2^{\lceil \log n \rceil}$ originating from itself. So the number of rounds needed for the leader to find out it won is bounded by:

$$2(1 + 2 + \cdots + 2^{\lceil \log n \rceil}) \leqslant 8n$$

At this point only $\lceil \frac{n}{2} \rceil$ rounds are needed for other nodes to receive the end message from the leader, so the number of rounds is clearly linear. □

Theorem 2. *Number of messages is $O(n \log n)$.*

Proof. A processor initiates a message with length 2^i only if it is not defeated by a processor within distance 2^{i-1}, so within any group of $2^{i-1} + 1$ processors only one will initiate a message with length 2^i. As a result of that initiation at most $4 \cdot 2^i$ messages will be passed, because every message is sent in two directions and will either travel 2^i edges and return or will be ignored within first 2^i steps. As mentioned before no message will have a length higher than $2^{\lceil \log n \rceil}$, so the total number of messages is bounded by:

$$4(1 \cdot n + 2 \cdot \lceil \frac{n}{2} \rceil + 4 \cdot \lceil \frac{n}{3} \rceil + \cdots + 2^i \cdot \lceil \frac{n}{2^{i-1} + 1} \rceil + \cdots + 2^{\lceil \log n \rceil} \cdot \lceil \frac{n}{2^{\lceil \log n \rceil - 1} + 1} \rceil)$$

Each of the $1 + \lceil \log n \rceil$ terms is bounded by $2n$, so the number of messages is not greater than $8(n + n \log n) \in O(n \log n)$. □

References

[1] D. S. Hirschberg, J. B. Sinclair, "Decentralized Extrema-Finding in Circular Configurations of Processors", *Commun. ACM* **1980**, *23*, 627–628, https://doi.org/10.1145/359024.359029.

[2] A. Iványi, "Leader election in synchronous networks", *Acta Universitatis Sapientiae Mathematica* **2014**, *5*, 54–1, https://doi.org/10.2478/ausm-2014-0005.