Franklin algorithm for leader election on synchronous, undirected ring

Marcin Wykpis

12 December 2024

1 Problem definition

Our goal is to elect exactly one leader node. The nodes are placed on a bidirectional ring connected by synchronous channels.

Here we analyze an algorithm proposed by Randolph Franklin [1] that sends $O(nlgn)$ messages.

2 Algorithm

At the beginning of the algorithm, each process sends messages containing its id to both neighbors. Each node can be active or inactive. At the beginning, we mark every node as active.

Let's consider an active node. In each round, the node receives both messages and compares their values with its own id. One of three cases is possible:

- One of the values is larger than the node's id. Then the process becomes inactive.
- Both values are smaller than the id. Then the process stays active and transmits its id to the neighbors.
- One of the values is equal to the id. The process is considered a leader and transmits *STOP* message to its neighbors

An inactive node simply retransmits received messages to the opposite channels until *STOP* is received.

3 Correctness

Let's consider two phases of this algorithm: before leader sends *STOP* and after the message is sent.

3.1 Before *STOP* message

- For the adjacent active nodes (connected by a series of inactive ones), at most one will stay active in the next round.
- Each active node received one message each from both adjacent active nodes
- Each node that stayed active transmitted exactly two messages
- If a node received its own id, it means that the value traveled through the whole ring and was compared (and considered larger) with every node in the ring. Only then the node sends *STOP* message and returns *LEADER*.

3.2 After *STOP* message

- At the end of previous round, only the leader node was considered to be active and it sent *STOP* to one of its neighbors.
- Each inactive node, starting from leader's immediate neighbors, receives a *STOP* message and retransmits it to the opposite node.

4 Time Complexity

Since at least one of each adjacent active nodes becomes inactive in each round, the number of rounds is at most $\lfloor lgn \rfloor$. In each round, there are exactly $2n$ messages sent. Additionally, we send $2n$ message before the first round and additional n *STOP* messages. In conclusion, the total number of messages sent is $O(nlgn)$, with at most $2nlgn + 3n$ messages.

5 Bonus: Gracefully Stopping

By adding an extra receive to the leader, we can guarantee that all channels are empty upon the end. The leader only has to receive a message from the opposite channel it sent the *STOP* message.

In each round before *STOP*, all sent messages are being received. In the *STOP* round, each inactive process receives and sends exactly two *STOP* messages. The leader process then receives the last two messages.

References

[1] Randolph Franklin. "On an improved algorithm for decentralized extrema finding in circular configurations of processors". In: *Commun. ACM* 25.5 (May 1982), pp. 336–337. ISSN: 0001-0782. DOI: 10.1145/358506.358517. URL: https://doi.org/10.1145/358506.358517.