Lab Session 4: Leader election in an anonymous network

Due date: June/03/2020

The objective of this lab session is to implement a leader election algorithm for anonymous undirected networks using a modification of the echo algorithm with extinction. We assume all nodes will get as input the number of nodes in the network. A process will learn about its neighbours by reading a configuration file with its neighbours IP addresses and port numbers. The name of this file will be passed as a command-line argument to the program followed by a positive integer N indicating the total number of nodes in the network.

Exercise:
Implement the following algorithm (from *Distributed Algorithms: An Intuitive Approach* by W. Fokkink) in *go*:

Initially, initiators are active at the start of round 0, and non-initiators are passive. Let process p be active. At the start of an election round $n \geq 0$, p randomly selects an ID $id_p \in \{1, \dots, N\}$, and starts a wave, tagged with n, id_p and a third parameter 0, $(n, id_p, 0)$. The third parameter indicates that this is not a message to its parent. This third parameter is used to report the subtree size in messages sent to a parent.

A process p, which is in a wave from round n with ID i, waits for a wave message to arrive, tagged with a round number n' and an ID j. When this happens, p acts as follows, depending on the parameter values in this message.

- If $n' > n$, or $n' = n$ and $j > i$, then p makes the sender its parent, changes to the wave in round n' with ID j, and treats the message accordingly.
- If $n' < n$, or $n = n$ and $j < i$, then p purges the message.
- If $n' = n$ and $j = i$, then p treats the message according to the echo algorithm but letting each message sent upward in the constructed tree report the size of its subtree; all other wave messages report 0.
- When a process is about to decide, meaning that its current wave n completes, it computes the size of the constructed tree. If this equals the network size N, then the process decides and becomes the leader. Otherwise, it moves to the next election round n+1, in which it randomly selects a new ID from $\{1, \dots, N\}$, and initiates a new wave.

Each line in the configuration file will have the following format:

```
<IP address>:<Port number>
```

except for the first line that will have the default local IP, a port, and an optional mark (*) indicating that the process is an initiator. An example of an input file is:

```
127.0.0.1:6002:*
10.80.29.90:6001
127.0.0.1:6001
```

The first line has the IP (always the local IP) and port where the process will accept TCP connections, and it also indicates that this process is an initiator. Non-initiators files will not have the ":" and the "*" at the end of the first line. Each time a process starts a new round it must print its random selected ID in the terminal. The process that is elected leader must print a message in the terminal indicating it is the leader and its ID.

Submitting your work for evaluation:

Create a zip file with all the relevant files needed to compile and run the exercise(s) together with a Readme.txt file. The file with the main package must be named **anon.go.** The Readme.txt file must include 1) the names of the team members, 2) a clear explanation of how to compile the program(s) (in case you decide to split your code into several packages), and 3) any information needed in order to test your programs. The evaluation will consider how easy is to use and test your programs as well as good programming practices such code documentation and organization.

Please name your zip file lab2_<name>_<last name>.zip and e-mail it to me (jorge.lobo@upf.edu) no later than **June/03/2020 at 16:30** to receive full credit. If you are working in a team submit only one zip file.