Lab Session 2: Basic Wave Algorithms

April/28/2019 – Due: May/05/2020

The objective of this lab session is to exercise the concept of waves. The implementation will be such that every processing node will run the same Go program; hence the wave algorithm can be started from any node and a parameter will be passed to the process indicating whether the node is an initiator. We will assume we have a stable network and communication between nodes will be through TCP. This means you can *assume* that all the neighbours of a node can eventually accept TCP connections. In other words, you don't need to (but you should) write robust code to handle failures caused by trying to start a TCP connection with a neighbour and always failing because, for example, the neighbour never starts, or that a neighbour fails in the middle of the distributed computation. A process will learn about its neighbours by reading a configuration file with its neighbours IP addresses and port numbers. The file will also contain the port number and the ID to be used by the process reading the file. The name of this file will be passed as a command-line argument to the program.

Exercise (wave algorithm):

- Implement the Echo algorithm discussed in class. Read below for the format of the configuration file. Each non-initiator process must print in the terminal the ID of the parent process. The process that decides must print a message in the terminal after the decision event and **send** a second way asking the processes to terminate. The initiator terminates when the second wave terminates. Each process will read the IPs and ports of the neighbours from a configuration file stored in the same directory where the program is located. 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 address, a port number, a **unique** ID (a positive integer) assigned to the process, and an optional mark "*" indicating that the process is the initiator. An example of the initiator's input file is:

  ```
  127.0.0.1:6002:3:*
  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, its ID 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. There will be a single configuration file with the mark "*".

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 for the Echo algorithm implementation must be named **echo.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 **May/05/2020 at 16:30** to receive full credit. If you are working in a team submit only one zip file.