CSC8103: Distributed Algorithms
Coursework Assignment

Paul Ezhilchelvan (`paul.ezhilchelvan@ncl.ac.uk`)

Aims: To improve your knowledge and understanding of Tree Wave and Tree Election distributed algorithms, and their implementation in Java.

Objectives

- Improve your understanding of distributed algorithms, in particular, Tree Wave and Tree Election algorithms.
- Demonstrate your ability to implement these algorithms in Java, following good programming practice.
- Develop your ability to test and evaluate distributed algorithms.

Tasks

Both tasks are based on the discussions we had during lectures on slides *1.26* and *2.9* (the right hand side columns).

1. Tree Wave Algorithm

Implement the Tree Wave algorithm in Java, following the guidelines provided below and also explained in the lectures.
Your program should produce clear console output, providing sufficient detail for you to debug the execution of the algorithm. You are **not** required to develop a GUI for your application.

Guidelines:

- Arrange the servers in a tree, and each server knows its *Neigh*.
- Run the algorithm for $N > 6$.
- To introduce non-determinism, number each process $0..N-1$; algorithm execution is made up of $100 \times N$ *iterations*.
- In each iteration, generate a random number $R \leq N$. Choose R processes randomly and allow only the chosen ones to execute the protocol. Token sending and receiving can be emulated by updating appropriate $rec[]$ bits.

- Ignoring the diffusion part, see how many processes decide in at least three different trees. If numbers are the same, is it a coincidence?

2. Tree Election Algorithm

Implement the Tree Election algorithm in Java, following the pseudo-code provided in your lecture handouts:
Your program should produce clear console output, providing sufficient detail for you to debug the execution of the algorithm. You are **not** required to develop a GUI for your application.

Guidelines:

- Arrange the servers in a tree, and each server knows its *Neigh*.
- Run the algorithm for $N > 6$.
- To introduce non-determinism, number each process $0..N-1$; algorithm execution is made up of $100 \times N$ *iterations*.
- In each iteration, as before, generate a random number $R \leq N$. Choose R processes randomly and allow only the chosen ones to execute the protocol.

Deliverables

Code

You are expected to submit all Java code and any third-party libraries (not including log4j and jUnit) required for us to run your code. You should also provide a brief README file detailing how to run your program and describe any configuration steps required.

Sample Console Output

You must include console output for the Tree Wave and Tree Election algorithms for each of the three tree types (unbalanced binary, balanced binary and arbitrary). You may include these in your submission as six separate text files. This sample output should be for the same tree structures as outlined in your written report.

Written Report

You should write a short report (no more than 500 words, excluding tables) to include the following:

- Describe the three trees (unbalanced binary, balanced binary, and arbitrary) used during the execution of the Tree Wave and Tree Election algorithms. (Nb. No marks are allocated for the graphical representation of these trees; a textual representation is fine).
- Tables detailing a number of executions of each algorithm for each tree type. These should include the number of rounds until termination, and the two deciding nodes for each execution.

- You must include an observations section commenting on the trends you see in the execution of each algorithm over the various tree types.

Marking Scheme

Assignment	**Percentage**
Tree Wave Algorithm	50%
Implementation	*35%*
Report	*15%*
Tree Election Algorithm	50%
Implementation	*35%*
Report	*15%*

Demonstration

In addition to your coursework submission, you will be expected to have demonstrated your code and your understanding of the algorithms ideally by **Thursday 25th October**, though very few demonstrations may be admitted on **Friday 26th October**, subject to time availability. Slot allocation sheets are available in Practical rooms. Please ensure you have signed up for a ten minute demonstration slot by the end of **Monday 22 October**.

Deadline and Submission

The **demonstrations** for both coursework must be done on or before **Friday 26th October 2018**.

Completed assignments including all source code, results and written reports must be submitted electronically through NESS by **Sunday 28th October 2018** at **23:15**.

This assignment is worth **20%** of your final mark for this module.

Questions?

If you have any further queries about your submission, you should approach a demonstrator during the practical sessions.

George Stamatiadis (`G.Stamatiadis2@newcastle.ac.uk`)
Adam Cattermole (`a.cattermole@newcastle.ac.uk`)
Pedro da Silva (`p.pinto-da-silva2@newcastle.ac.uk`)