Project One CDD101: Concurrency

Joseph Kehoe

November 6, 2017

1 Wator Simulation

You are required to implement the Wator simulation. The full specification of this simulation is provided in the accompanying document from the "Mathematical Recreations column". The simulation will take 7 parameters:

NumShark: Starting population of sharks;

NumFish: Starting population of fish;

FishBreed: Number of time units that pass before a fish can reproduce;

SharkBreed: Number of time units that must pass before a shark can reproduce;

Starve: Period of time a shark can go without food before dying;

GridSize: Dimensions of world;

Threads: Number of threads to use.

The project will be completed in teams with two people in each team. Development should be undertaken using pair programming. There are four stages in the project with each stage worth equal marks.

1. Serial Implementation;
2. Concurrent Implementation;
3. Benchmarking;
4. Optimisation.

Each team will perform the first two stages on their own code. For the remaining two stages each team will benchmark and optimise another team's code (chosen at random).

1.1 Serial Implementation Friday 17th November

A full serial implementation of the Wator simulation. Output can be ASCII based or graphical.

1.2 Concurrent Implementation Friday 24th November

OpenMP will be used to turn the serial version into a parallel version. The parellel version must give the correct outputs but shoulod not be optimised.

1.3 Benchmarking Friday 1st December

Benchmarks will be produced that show how the program scales and measures its efficiency. The output of this phase is a short report describing the benchmarks used and showing the results obtained.

1.4 Optimisations Friday 8th December

Based on the benchmarking report, the code will be optimised to improve its performance. The optimised code should be benchmarked and the results compared to the previous unoptimised version.

2 Implementation Tools

- The project must be implemented in C++ on Linux using the g++ compiler and emacs (or vi);
- Documention must be produced using Doxygen;
- A Makefile must be provied that compiles the project (debug and release versions), performs a clean up and creates the documentation by running Doxygen;
- The projects must be stored on Github - one github project for each team.