Coursework 3: Life
Student Report

4CCS1PPA/APPA Programming Practice and Applications

Ching Lok Tsang, Hemchandra Erukulla , Ruijie Li

ching.l.tsang@kcl.ac.uk, hemchandra.erukulla@kcl.ac.uk , ruijie.1.li@kcl.ac.uk

21/02/2024

INTRODUCTION – Distribution of tasks

In this project, Ching, Hemchandra and Ruijie have collaborated to create "Game of Life" which is a cellular automaton invented by Cambridge mathematician John Conway.[1] We have evenly decided on the workload for each individual to complete in both the base and challenge tasks. For instance, Hemchandra has been tasked to create the simplest form of life: "Mycoplasma", whilst Hemchandra and Ruijie are creating "Yersinia" and "Carsonella". In addition, Hemchandra has completed the "non-deterministic cells" and "Buttons implementation" challenges, with Ching and Ruijie finishing the "Symbiosis" and "Disease" challenge tasks.

LIFEFORMS– Mycoplasma, Yersinia, Carsonella, Deterministica

1. Hemchandra has created **Mycoplasma**, complying with the rule set given where:

- If the cell has fewer than two live neighbours it will die
- If the cell has two or three live neighbours it will live on to the next generation
- If the cell has more than three neighbours it will die
- Lastly, any dead cell will come alive if it has exactly three neighbours

Mycoplasma cells are set with the colour blue as can be seen on figure 1.

2. The **Yersinia** lifeform has been created by Ching, incorporating different colours (yellow, crimson, dark goldenrod as can be seen in figure 1) into it. The rule set is for Yersinia is:

- If the cell has two or more live neighbours with the same colour, then the cell dies
- Otherwise, it will live onto the next generation with the cell changing to another random colour
- However, if cell is dead and has exactly 2 neighbours, then that cell will become alive with a random colour

3. Ruijie created **Carsonella**, implementing changes in its behaviour as time progresses in which the rules are as follows:

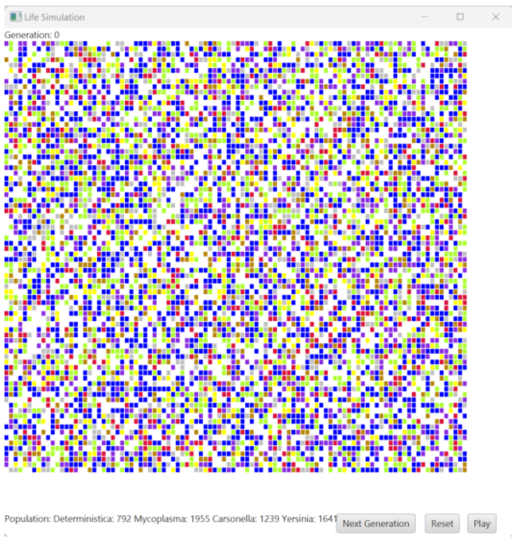
- When the cell is first "born" it has a death probability of 10%
- Every generation, it's "Age" increases by one year
- As its age increases, the probability of death increments by 10% per year

Carsonella is also impacted by a pathogen (as per the challenge task described later), where **Mycoplasma** is the benefactor while **Carsonella** suffers. Carsonella can be seen in the "GreenYellow" colour which is visible as a light green on figure 1. When the Carsonella cell is infected by a pathogen (as per the challenge task described later) it is shown in a "Dark Green".

4. Hemchandra created **Deterministica** as per the challenge task, which will be described later on in the report, the Deterministica cells can be seen in purple on the grid below.

[1] John Conway (n.d.). *Play John Conway's Game of Life*. [online] playgameoflife.com. Available at: https://playgameoflife.com/info

Figure 1 – Grid for the Game of Life with Mycoplasma, Yersinia, Carsonella and Deterministica



```
Generation: 0
Population: Deterministica: 792 Mycoplasma: 1955 Carsonella: 1239 Yersinia: 1641
```

<u>CHALLENGE TASK – Non-deterministic cells</u>

Hemchandra created the class titled "Determinsitica", which forms the blueprint for the cells which implement a "non-deterministic" behaviour. Listed below are the probability-based rules for the Deterministica lifeform.

• **RULE 1 (25% Chance)**: If less than three alive neighbours, replaces all dead neighbours with new Deterministica cells and current cell dies

• **RULE 2 (35% Chance)**: Count the frequency of lifeform types of surrounding cells, most common lifeform will "take over" the Deterministica cell and occupy its place on the grid.

• **RULE 3 (5% Chance)**: The cell does nothing, remains as is.

• **RULE 4 (35% Chance)**: The cell dies (as if it was a random death)

This task has been completed by using a random number generator, which generates a double between 0 and 1, if the generated double falls between the probability values of a given rule, then upon the "**act()**" method being called, the given rule would be executed. The most challenging rule to incorporate was rule 2, this has been done by iterating through all living neighbours of a cell, the "**instanceof**" operator was used to identify the type of each cell and thus increment their relevant counters. The largest value of the counters was identified, and a cell of that type would take the place of the existing "Determinsitica" cell.

CHALLENGE TASK – Symbiosis

Ching has been assigned to this task, where he decided that the symbiotic relationship between Mycoplasma and Carsonella to be parasitism. This is when the symbiont (Mycoplasma) resides on the host (Carsonella),[2] transmitting pathogens to the host and consequently harming the host, possibly even causing its death due to diseases. On the other hand, Mycoplasma benefits from feeding on Carsonella. The rule set is shown below:

• If a Carsonella cell is not infected and has a Mycoplasma neighbour, it has a 20% chance to get infected by Mycoplasma, transmitting pathogens to the cell.

• If a Carsonella cell harbours pathogens, it has a 10% chance to die from diseases caused by the pathogens.

• If a Mycoplasma cell has an infected Carsonella neighbour, an additional rule is introduced where if the cell has exactly four neighbours, it will live onto the next generation, thus proving advantageous to the Mycoplasma.

An infected Carsonella cell would change from a light green to a dark green colour. To complete this task, an algorithm was implemented. When the "act()" method is called on any Carsonella cell, the neighbouring alive cells are iterated through, to identify if one of the neighbouring cells is a mycoplasma lifeform. If a neighbouring mycoplasma is found, then a random double generator produces a number, if this number is lower than 0.2, then the Carsonella cell becomes infected.

CHALLENGE TASK - Disease

This challenge task was completed by Ruijie where there is a disease characterized with the three elements:

• When a cell is initialized, it has a 15% chance of being infected.

• The disease spreads from one cell to its neighbours.

• And once in a diseased state, the cell's behaviour changes.

Ruijie implemented these elements where:

• If one cell is infected with the disease, there is a 3% chance for the disease to spread to any of its neighbours.

• And when a living Mycoplasma cell is in a disease state, the rules change where:

  • If the cell has four or five neighbours, it will live on to the next generation. Otherwise, the cell dies.

  • Also, when a dead cell is in a diseased state and it has exactly two neighbours, it will come alive.

• When a living Yersinia cell is in a disease state, the rule set is altered where:

  • If the cell has no neighbours of the same colour, it will die.

  • Additionally, when a dead cell is in a diseased state and has exactly four neighbours, it will come alive.

• Lastly, when a living Carsonella cell is in a disease state, the rules are changed such that:

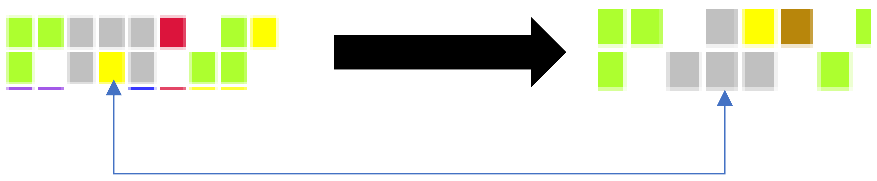
  • The probability of death of the cell increases by 20% percent per generation (year of age) rather than 10% otherwise

  • When a dead cell is in an infected state, the combined age of the surrounding cells must now be less than 20, for the cell to come back to life rather than 30 as previously, thus making the requirements to come back to life more stringent.

[2] Overstreet, R.M. and Lotz, J.M. (2016). Host–Symbiont Relationships: Understanding the Change from Guest to Pest. *The Rasputin Effect: When Commensals and Symbionts Become Parasitic*, [online] 3, pp.27–64. doi:https://doi.org/10.1007/978-3-319-28170-4_2.

Upon becoming infected by the disease, any cell will have it's colour changed to "silver", as can be seen in the figure below. The yellow Yersinia cell has become infected by its neighbours, and its colour has changed to silver to indicate this (shown by the blue arrow). Upon becoming infected, cells no longer behave as described in the "act()" method, instead, they now behave as described in the "onInfection()" method, which contains stricter rules, emulating the difficulty of living with the disease.

Figure 2 – a cell in a diseased state, infecting its neighbours.



CHALLENGE TASK – "Next generation", "Play/Pause", "Reset" button implementation.

Ruijie and Ching has suggested the implementation of interactive buttons with different functions, and Hemchandra successfully implemented them. There are three buttons: a "Next generation" button, an "Play/Pause" button, and a "Reset" button, where:

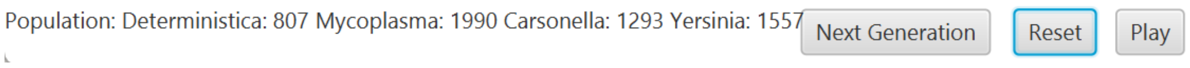
• The "Next generation" would progress the grid into the next generation, carrying forward all the changes according to the rule sets for each lifeform and the wider factors such as disease and symbiosis.

• The "Play/ Pause" button can be toggled so that the grid would automatically move to the next generation in regular time intervals.

• The "Reset" button would reset all the cells on the grid and return the grid back to zero generations.

This task has been done by adding additional buttons to the "SimulatorView" class, the hardest part to implement was the play/pause button. An alternating Boolean variable was created, which keeps track of whether the simulation is currently running automatically. Upon pressing the button, the text label would also alternate between "play" and "pause" to clearly display the function of the button. Upon pressing the play button, a new thread was initiated within which the simulation could run.

Figure 3 – "Next generation", "Play/Pause", "Reset" buttons on the grid


Population: Deterministica: 807 Mycoplasma: 1990 Carsonella: 1293 Yersinia: 1557