Modelling Predator-Prey Dynamics within Conway's Game of Life

Topic: Swarm Intelligence/Collective behaviour/Self organization and self-assembly

- Implemented the Game of Life using Python

  GitHub Repository with README available here:
  https://github.com/sc19sgs/CGOL_Bio_Inspired/blob/main/README.md

  To run this program from a python virtual environment

  1.) Activate your virtual environment

  2.) Run pip install -r requirements.txt to install the necessary dependencies

  3.) Run python Game_Mode.py to start the application and choose your Game Mode

- Implemented and experimented with alterations of Conway's Game of Life

Sina Ghanbari Saheli, Daniel Lartey, Chemseddine-Wassim Benimoussa

Sc19sgs, Sc20dl, Sc20cwb

Introduction

This report explores self-organisation & swarm intelligence in Conway's Game of Life (GOL). Swarm intelligence is the collective behaviour of decentralised, self-organised systems **(Wikipedia Contributors, 2019)**. This is inspired from biological systems in nature where agents follow simple rules which can lead to intelligent behaviour. The GOL was created by John Conway in 1970, and it describes basic rules for cells to evolve based on its initial state. The GOL runs for an infinite number of time steps based on the initial state, where each cell can comprise of two states; alive or dead **(Gardner, 1970)**.

At each time step, each cell interacts with its 8 diagonally connected neighbours to update the state for that cell with the following rules:

1. A live cell with less than 2 live neighbours dies (underpopulation)
2. A live cell with 2 or 3 neighbours remains unchanged
3. A live cell with more than 3 live neighbours dies (overpopulation)
4. A dead cell with 3 live neighbours becomes a live cell (reproduction)

From a given initial state of the cells, there are various patterns which can be shown, for example the **still life, oscillator and spaceship** patterns.

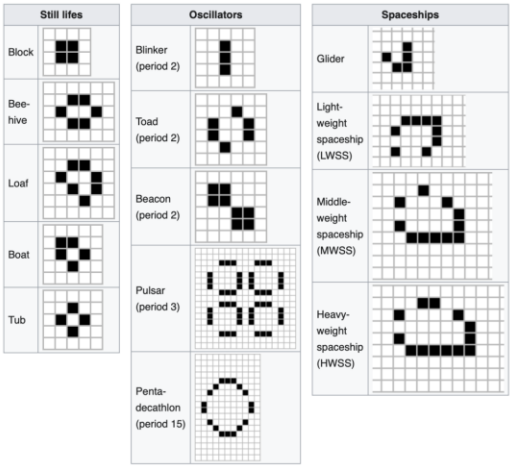

Figure 1: Examples of patterns within the GOL (Lefebvre, 2022)

Still Life Pattern

The still life pattern is one in which the cells don't change between generations. There are two main subgroups of the still life pattern – **strict still lifes and pseudo still lifes**. The main distinction is that a pseudo still life can be partitioned into 2 or more islands which themselves are still lifes. For a strict still life, removing one of the islands will destroy the stability of the pattern **(Wikipedia, 2023b)**.

Oscillator Pattern

The oscillator pattern repeats after a set number of generations. The period of the pattern is the time from the first time the pattern appears to the next time. An oscillator consists of two parts. The **rotor** (cells which oscillate) and the **stator** (cells which remain alive during the whole period). There are many oscillator states, with the smallest being the blinker at period 2 requiring 3 cells, and the largest being the capped glider gun with period 856 requiring 92 cells. Figure 1 shows an example of the blinker oscillator pattern **(LifeWiki, 2024)**.

Spaceship Pattern

The spaceship pattern reappears after a fixed number of timesteps at a different location within the grid. The difference between this pattern and the oscillator pattern is the emphasis on the reappearance in a different location within the grid. Some spaceship patterns can move at faster speeds than others.

The GOL conforms to many bio-inspired concepts, such as swarm evolution, stigmergy and clustering. Since there isn't a central leader within the GOL, the cells must organise themselves without an organiser **(Garnier et al., 2007)**. Stigmergy is also exhibited as the cells indirectly coordinate in the environment based on the rules. There isn't any planning or direct communication between the cells in the GOL. Each of the cells follow the rules. Finally, clustering may/may not be exhibited based on the initial state, since the cells may either diverge or form a cluster

Now that a basic overview has been presented regarding CGOL, the next section will focus more on the background research regarding the reasoning of the rules, advancements made since the introduction of CGOL and an analysis of why CGOL may/may not be a good representation of swarm intelligence & self-organisation.

Background Research

There are many applications which build upon the GOL. One application uses qubits in each cell which evolve by repeated application of birth & death. In this model, a species is characterised by two parameters – a preferred neighbourhood of liveness (tendency to herd) and a resilience parameter (vulnerability to environmental changes). In this model it mimics environmental catastrophes, and the predator-prey model qualitatively describes strongly & weakly coupled predator-prey systems **(Faux and Bassom, 2023)**.

Furthermore, CGOL has been extended to implement digital logic using patterns such as the glider pattern to create elementary logic gates such as NOT, AND OR and a 4-bit adder. The motivation by Carlini to implement these gates using CGOL was to experiment whether it's possible to emulate an intel 4004 microprocessor on top of CGOL, as seen in Figure 2 **(Carlini, 2021)**. These applications of CGOL in various domains illustrates that the rules are not just limited to simulating

how a cell may survive/die from one state to another. There is clearly swarm intelligence and collective behaviour between these cells to form building blocks of a larger system. Finally, there have been attempts to model CGOL in 3 dimensions, but for the scope of this report we have not considered this **(Bays, 1987)**.

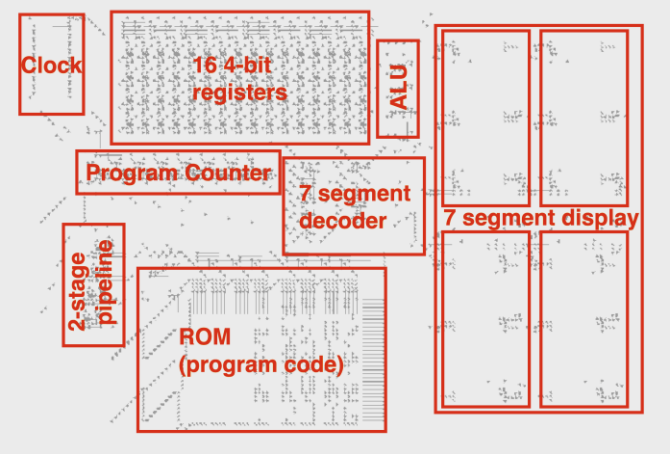

Figure 2: Example of CPU in CGOL using cellular automation (Carlini, 2021)

It can be argued that CGOL doesn't adhere to the principles of swarm intelligence, since it doesn't follow any of the traditional models. For example, CGOL doesn't show alignment or cohesion in the Boids model.

Adaptation of traditional rules

There are many variations the GOL. For example, Highlife has an additional rule where a dead cell with 6 live neighbours comes to life. This new rule leads to the replicator pattern, as seen i**n Figure 3**, which reproduces themselves on a diagonal line. When two replicators try to expand into each other, the pattern in the middle vanishes **(Wikipedia, 2023)**.

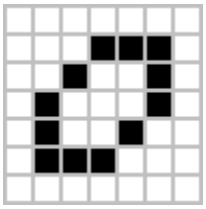

Figure 3: Replicator pattern (Wikipedia, 2023)

Another variant is Brian's Brain, which introduces a third state for cells – dying. The rules for a dying cell are as follows. A dying cell turns on in the next generation, while live cells in die and cells in the off state remain unchanged. Since a dying cell leads to directional movement, most of the patterns observed in Brian's brain are spaceships. In addition, the spaceships emit other spaceships and patterns can explode messily and chaotically **(Wikipedia Contributors, 2022)**.

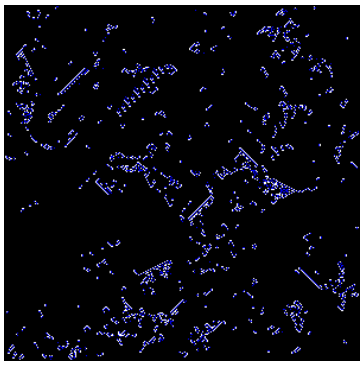

Figure 4: Example of chaotic nature in Brian's Brain (LifeWiki, 2020)

From the background research, we were motivated to implement CGOL and build additional rules to observe and evaluate how this affects the stability of the cells, collective behaviour and the swarm intelligence. The next section explores our approach to the modification of the rules, along with a justification and prediction as to how this will affect the mechanics of CGOL.

Methodology

We experimented with the rules in the GOL to model a scenario with wolves as the predator, and mice as the prey. These animals conformed to the original rules, but with some minor differences.

Rule Modification

The different modes we created can be accessed from the Game_Modes.py window, each mode being an improvement of the previous one by adding new rules and concepts.

Mode 1: Conway's Game of Life (Predator and Prey)

This mode uses two distinct species. Wolves as predators and mice as prey. Both species follows the basic GOL rules, but if a mouse has exactly 3 wolves in its neighbourhood it will die, and a new wolf is created at that position, in order for the predator to take precedence. Overall, this imitates a predator being able to eat the prey. We also made the cells in the grid wrap around continuously, to eliminate boundaries.

Mode 2: Conway's Game of Life (Food Supply)

This mode is an improvement on Mode 1. It keeps the same configurations as the basic one with an extra set of rules:

- Relative Population bars for the mice and wolves which acts as our fitness function.
- Mice rewarded for eating cheese by spawning a new group of mice on the grid
- Wolves rewarded for eating a mouse by spawning a new group of wolves on the grid
- Preventing overpopulation by killing excess wolves which take more than 50% of the grid, killing the wolves at random.

Mode 3: Conway's Game of Life (Infection)

The infection mode adds a virus in addition to the rules in Mode 2, which tries to balance the population. It targets the species which is overpopulated; should the relative population of a species be greater than 70% - then the virus will target this species. Once the relative population is below 70%, then the virus will disappear.

Mode 4: Conway's Game of Life (Infection Endless)

This mode builds upon the foundation provided in Mode 3, however the population of any species can't be 0. This results in infinite predator-prey dynamics, since a minimum of 4 of each species should be present on the grid. Should an extinction event occur, a "glider" pattern spawns for the affected species, aiming to repopulate the grid.

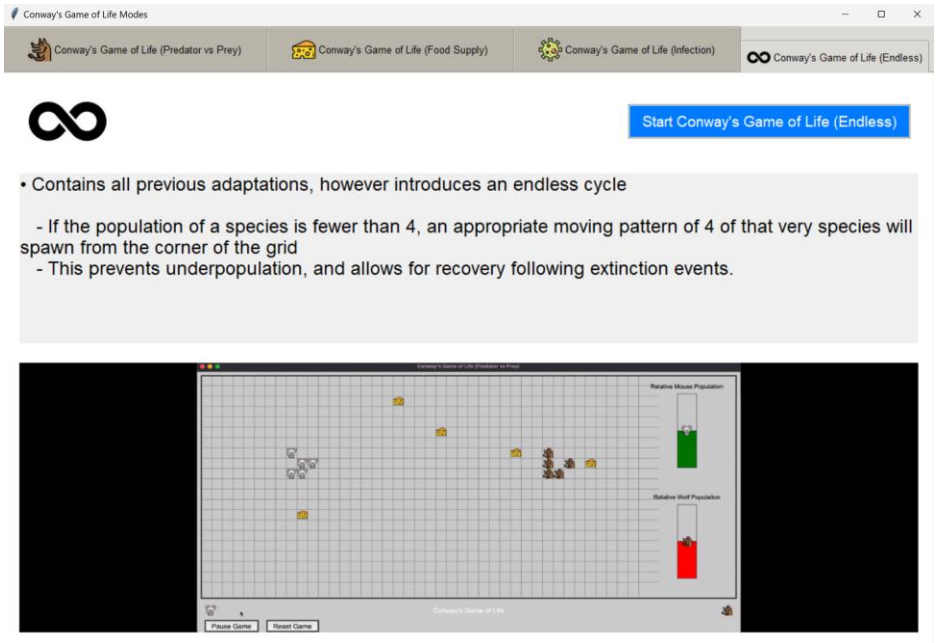

Figure 5: Diagram showing application with rule modifications

Reasoning behind the rules

We modified the rules to try and get the wolves and mice to coevolve. Coevolution occurs when two or more species affect each other's evolution. As one species starts to improve, the other one becomes objectively worse and must improve to stay competitive. We believe that these rules would allow coevolution as:

- Wolves eating mice will force the mice to improve to survive and penalising wolves for not eating mice will force them to pursue mice to remain alive.
- Mice eating cheese provides maintainability for the mice to stay alive & are penalised for not doing so.
- Preventing overpopulation will allow the wolves and mice to coevolve, as one population will not become too dominant, by using the virus.

- We introduced a random stable population for the wolve and mice when they eat their food sources, as this will release pheromones acting as a signal to reproduce.

Metrics

We track the number of mice & wolves and the ratio of mice to cheese to wolves throughout our simulation. These metrics can estimate whether coevolution is occurring and how stable/chaotic the modified GOL is. This can be tracked via the metrics we collect with the population of the wolves and mice. For coevolution to occur, the fitness function should exhibit an inverse relationship (i.e. when the relative population of the wolves increases, the relative population of the mice decreases and vice versa).

Results

Game Mode 1 (Simple CGOL with Predator vs Prey)

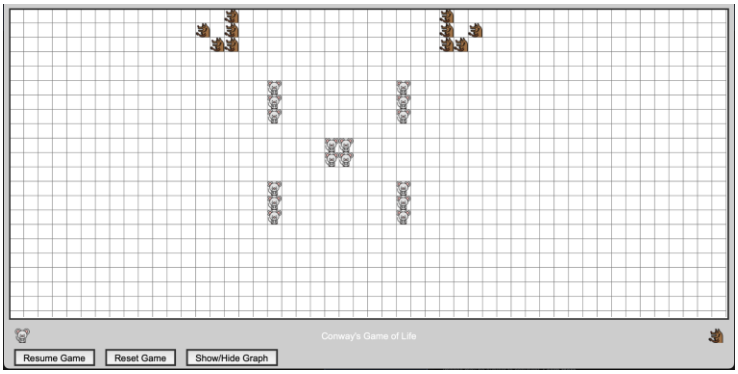

Figure 6: Diagram showing initial Encroachment configuration of Game Mode 1

The Encroachment setup includes wolves placed in a 'glider' pattern configuration, aimed at converging towards the centre where stable populations of mice are located. This setup was chosen as an ideal demonstration for observing how localised initial predator configurations affect spatially isolated prey groups.

This led to the wolves eliminating the nearest oscillating groups of mice and destroying the central stable population. Their congregation of the two wolf populations caused a settling into 6 stable oscillating patterns. The remaining mice groups, located at the bottom, weren't reached by the wolves and thus survived. This replicable outcome demonstrates the impact of spatial separation on survival rates in Predator-Prey systems, mirroring ecological scenarios where isolation can sometimes protect prey from predators.

Population Dynamics Graph

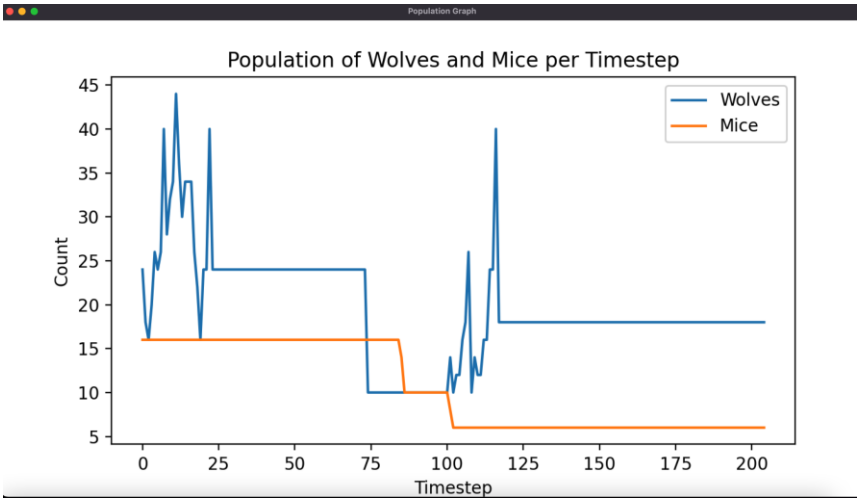

Figure 7: Graph for population at each timestep

Figure 7 shows significant fluctuations in the wolf population, correlating to the nature of their movement in packs in the 'glider' formation, where the population must constantly shift for perceived 'movement' to take place. The sudden decreases in mice population, on account of the wolf attacks can be witnessed at time step 80 onwards, with another unstable spike in the wolf population occurring shortly after due to the gathering of wolves, resulting in the GOL rules being applied numerous times before a stable predator population is developed.

Game Modes 3 & 4 (CGOL Infection + Endless Mode)

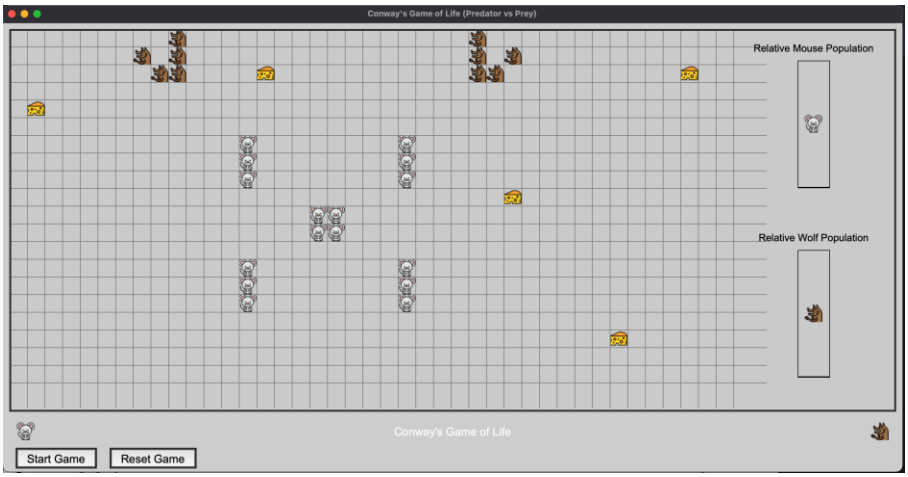

Figure 8: Diagram showing *Encroachment* initial configuration of Endless Game Mode

Again, in the Encroachment starting formation, wolves and mice are placed to align wolves in a way that encourages their movement towards a densely populated area of stable mice.

Due to the arbitrary placement of cheese blocks, multiple tests resulted in a variety of results for the normal 'Infection Game Mode', including:

- The predators rapidly converging into stable patterns, without outnumbering the prey by enough to trigger a virus attack.
- The oscillating prey patterns encountering the cheese resulting in them being rewarded with "mobile" mice patterns being spawned. This stemmed into the wolves encountering more mice, which similarly caused the predators to be rewarded with "mobile" wolf patterns spawning. The overpopulation of the wolves would lead to the infection often wiping out the remaining species, resulting in extinction.

This time around in the 'Endless Game Mode', extinction is non-viable, meaning the hinted back-and-forth race between the two species should be sustained for longer. This was indeed the case as Figure 9 suggest a mimicry of co-evolution dynamics. In true biological coevolution the changes in one species genetically influence the evolutionary adaptations in the other, however in this simulation, while the wolves and mice respond to each other's populations and the available resources, their genetic traits (in this case being rules of survival and replication), remain fixed unless we manually adjust them in the code. This is potentially where a genetic algorithm could be useful.

The results displayed in the plots in Figure 9 provide insights into these dynamics, showcasing the fluctuating populations of wolves and mice over the generations, which are influenced due to factors such as food supply and disease spread. Immediately it can be seen that it takes a significantly longer time for a stable population to be reached for both species, in comparison to previous Game Modes. Specifically in Figure 7, 125 timesteps provides convergence towards a stable eco-system, however in the Endless game mode almost 1500 timesteps are required for stabilisation to occur. This is naturally due to the more intricate and adaptive interactions allowed by the simulation's rules in this mode.

Population Dynamics Graph

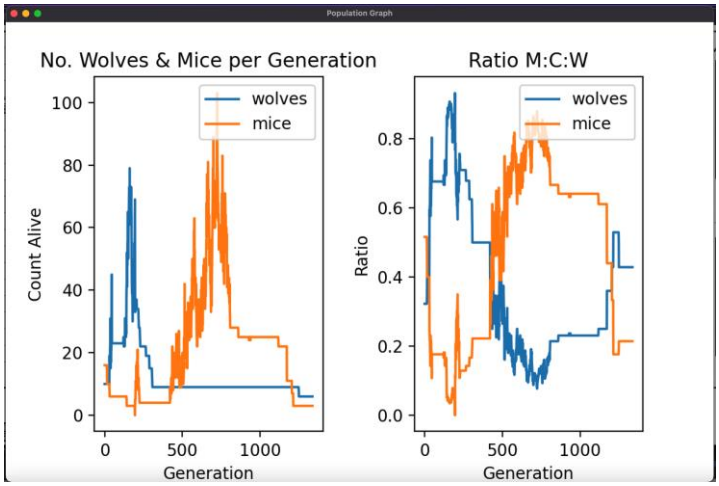

Figure 9: Diagram showing *Encroachment* result plots for Endless Game Mode

Conclusion

From our hypothesis which was to evaluate whether we could show evidence of coevolution within the modified GOL, we can see that our results show that for Game Mode 4, there is evidence that it may exists. This is due to the metrics, which can be seen in Figure 9, illustrating the inverse relationship between the ratio of mice to cheese to wolves. As the population of the mice increases, the population of the wolves decreases from 0-250 generations. From 250 generations onwards, the number of mice increases until 750 generations, with slight fluctuations indicating that the wolves are evolving. Also, from 250 generations onwards, the population of wolves stabilises, indicating that it's not improving at all over time. However, since the number of mice drops below this stable wolf population after 1250 generations, we can justify that it may be evolving, just not as fast as it previously did. These population fluctuations suggest ongoing adaptive responses, but despite this, it's important to note that these dynamics, while mimicking coevolutionary interactions, are not definitively indicative of true coevolution, because they are largely the product of our predefined simulation rules, rather than adaptions from an evolutionary process.

In contrast Game Mode 1 did not exhibit any patterns consistent with coevolution whatsoever. From the population of wolves and mice in Figure 7, we can see that the population of mice is strictly monotonically decreasing, without reciprocal adaptive changes from the wolves – insinuating that there are no coevolutionary dynamics present in this simpler game mode.

One way we could further improve this implementation is by including a genetic algorithm using neural networks which would allow both species to select the best traits from each population, proving a more realistic implementation of how coevolution should work in practice. Our interest particularly lies in embedding a learning component within each species' reward mechanism, following the capture of a food source. Currently, when a reward is triggered, the "mobile" pattern that is spawned is that of a "glider" pattern, the simplest moving pattern in Conway's Game of Life.

However, the game features various other patterns that enable movement in unique ways, affecting factors such as the size of the moving herd or the direction of movement. By integrating a genetic algorithm, both predators and prey could experiment with different movement patterns as rewards originating from diverse grid areas, to discover the most advantageous strategies to maximise population and avoid the infection. This process could be refined by employing a learning algorithm specific to each species, which would take the immediate population changes following each reward as an input. Due to the time constraints, we didn't feel it was necessary to implement this, as we did not want to deviate too far from the original rules.

Bibliography

1. Bays, C. 1987. Candidates for the Game of Life in Three Dimensions'. Complex Systems. 1, pp.373–400
2. Carlini, N. 2021. Improved Logic Gates on Conway's Game of Life - Part 3. nicholas.carlini.com. [Online]. [Accessed 4 May 2024]. Available from: https://nicholas.carlini.com/writing/2021/improved-logic-gates-game-of-life.html
3. Faux, D.A. and Bassom, P. 2023. The game of life as a species model. American Journal of Physics. 91(7), pp.561–561
4. Gardner, M. 1970. MATHEMATICAL GAMES. Scientific American. 223(4), pp.120–123
5. Garnier, S., Gautrais, J. and Theraulaz, G. 2007. The biological principles of swarm intelligence. Swarm Intelligence. 1(1), pp.3–31
6. Lefebvre, P. 2022. Conway's Game of Life – Xojo Programming Blog. Xojo. [Online]. [Accessed 4 May 2024]. Available from: https://blog.xojo.com/2022/05/11/conways-game-of-life/
7. LifeWiki 2020. OCA:Brian's Brain - LifeWiki. Conwaylife.com. [Online]. [Accessed 4 May 2024]. Available from: https://conwaylife.com/wiki/OCA:Brian%27s_Brain
8. LifeWiki 2024. Oscillator - LifeWiki. conwaylife.com. [Online]. [Accessed 9 May 2024]. Available from: https://conwaylife.com/wiki/Oscillator.
9. Wikipedia 2023a. Highlife (cellular automaton). Wikipedia. [Online]. [Accessed 4 May 2024]. Available from: https://en.wikipedia.org/wiki/Highlife_(cellular_automaton).
10. Wikipedia 2023b. Still life (cellular automaton). Wikipedia. [Online]. [Accessed 9 May 2024]. Available from: https://en.wikipedia.org/wiki/Still_life_(cellular_automaton).
11. Wikipedia Contributors 2022. Brian's Brain. Wikipedia. [Online]. Available from: https://en.wikipedia.org/wiki/Brian%27s_Brain.
12. Wikipedia Contributors 2019. Swarm intelligence. Wikipedia. [Online]. [Accessed 9 May 2024]. Available from: https://en.wikipedia.org/wiki/Swarm_intelligence.

Statement of Contributions

- **Report**: Sina Ghanbari Saheli, Daniel Lartey, Chemseddine-Wasim Benimoussa
- **Codebase**: Sina Ghanbari Saheli, Daniel Lartey, Chemseddine-Wasim Benimoussa