Selective Breeding of Cellular Automata using Monochrome and Colourised Games of Life

Author: Peter D. Turney, peter.turney@gmail.com

Submitted to: Artificial Life, MIT Press

Category: Article reporting original research

Date: June 25, 2024

Abstract

The Game of Life is a universal computer, capable of computing anything a Turing machine can compute. A program in Life is written in a two-dimensional grid space, by setting the colour of each square in the grid to either white or black. With more than 50 years of experience, Life programmers have created a large collection of two-dimensional configurations of black and white squares. Each configuration of squares serves as a module that can perform a useful computation. In this article, we propose an alternative approach to the Game of Life. Instead of *programming* Life, we use an evolutionary algorithm to *breed* Life. We use a three-part approach, with (1) a population of small seed patterns, (2) a single large target pattern, and (3) an evolutionary algorithm. The population of seed patterns is initially random. A seed is randomly selected from the population and allowed to grow, and then it is assigned a fitness score, based on how well it matches with the large target pattern. The evolutionary algorithm makes small, random changes to the seed population, selecting seeds that increase the fitness of the match with the target pattern, and removing seeds that have low fitness. We demonstrate some success with breeding as an alternative to programming, in the Game of Life (monochrome black and white) and in the Immigration Game (colourised white, red, and blue). Breeding yields structures that appear to be more organic and lifelike than programmed structures.

Keywords: cellular automata, evolutionary algorithms, Game of Life, breeding automata, colourised Life.

1. Introduction

Cellular automata began with the work of Stanislaw Ulam and John von Neumann in the 1940s (Poundstone, 2013). A cellular automaton is a regular grid of cells, typically square cells in two-dimensional space. Each cell can be in a finite number of different states, which are usually distinguished by colours or numbers. The grid is initialized by setting the starting states for each cell. The states of the cells change at

integer intervals according to a given set of rules. All of the cells at time t simultaneously change their states at time $t + 1$. The new state for a given cell at time $t + 1$ is a function of the states of the cell's neighbours in the grid at time t.

The most popular cellular automaton is the *Game of Life*, introduced by John Horton Conway in 1970 (Gardner, 1970). At first, the game was played manually, on a Go board. When computers became common, the Go board was replaced with a computer screen, displaying a grid of black and white squares. At the beginning of a game, the squares are all white. The solo human player then creates a pattern by changing some of the white squares to black squares. The computer then animates the screen with a sequence of colour changes, from black to white and from white to black, following a set of deterministic rules. The objective of the game is to create an interesting sequence of images. As the game runs, black shapes grow and shrink. Some black shapes crawl across the screen, traveling in a cyclic sequence of motions.

The Game of Life appears to be simple, but it has been proven that the game is a universal computer (Berlekamp *et al.,* 1982). That is, anything that can be computed with a standard digital computer can be computed in the Game of Life. Enthusiasts have spent more than 50 years exploring the possibilities of the Game of Life. This exploration involves much trial and error, with a slow accumulation of expertise and experience. There are some software tools that support this research, but the most interesting discoveries generally come from hard work and practice. Johnston and Greene (2022) provide an excellent introduction to the discoveries that Game of Life enthusiasts have made in the last half-century.

A difficulty we face with the Game of Life is that a small change to a Game of Life seed pattern can result in a large change in how the seed grows. In general, there is no easy way to direct the growing pattern to a desired outcome. This is where an evolutionary algorithm can be helpful. Consider that an acorn is quite different from a fully grown oak tree, yet evolution is able to create a path for growth that begins with an acorn and ends with an oak tree.

The general approach of evolutionary algorithms is to make small random changes to a structure and measure the impact of these changes. If a change improves the performance of the structure, the change is kept, although it might be dropped later. If a change reduces the performance of the structure, the change is removed, although it might be restored later. The structures we manipulate in this article are Game of Life patterns.

Evolutionary algorithms can achieve results that are similar to the constructions made by Game of Life enthusiasts, yet evolutionary algorithms do not reason like humans. Just as biological evolution has created extremely complex and capable organisms without planning and reasoning, evolutionary algorithms can create complex computational structures and patterns without planning and reasoning.

In Section 2, we introduce Conway's *Game of Life* and Woods' *Immigration Game*. These two cellular automata are among the earliest investigated cellular automata. Section 3 explains how evolutionary

algorithms can be used to evolve patterns in cellular automata. The aim is to reduce the human effort required to create interesting patterns, by sketching the desired result and allowing evolution to search for a solution. Section 4 shows the results of our experiments with evolving cellular automata. Section 5 discusses related work, Section 6 considers future work, and Section 7 summarizes the conclusions.

2. Cellular Automata

There are many different kinds of cellular automata. We will describe two closely related cellular automata, John Conway's *Game of Life* (Gardner, 1970) and the *Immigration Game*, created by Don Woods (Wainwright, 1971). The Game of Life uses two colours, black and white, whereas the Immigration Game uses three colours, white, red, and blue. If colour differences are ignored, the two games are the same.

2.1 Conway's Game of Life

Martin Gardner (1970) presented John Conway's *Game of Life* as three desiderata and three rules. The desiderata are as follows (Gardner, 1970, p. 120):

1. There should be no initial pattern for which there is a simple proof that the population can grow without limit.
2. There should be initial patterns that *apparently* do grow without limit.
3. There should be simple initial patterns that grow and change for a considerable period of time before coming to end in three possible ways: fading away completely (from overcrowding or from becoming too sparse), settling into a stable configuration that remains unchanged thereafter, or entering an oscillating phase in which they repeat an endless cycle of two or more periods.

The three rules are as follows (Gardner, 1970, p. 120):

1. Survivals. Every counter (playing piece) with two or three neighboring counters survives for the next generation.
2. Deaths. Each counter with four or more neighbors dies (is removed) from overpopulation. Every counter with one neighbor or none dies from isolation.
3. Births. Each empty cell adjacent to exactly three neighbors—no more, no fewer—is a birth cell. A counter is placed on it at the next move.

Our terminology is different from Gardner's terminology. Gardner's *counter* corresponds to our *live black cell* and Gardner's *empty cell* corresponds to our *dead white cell*. This change in terminology comes from abandoning the Go board and switching to the computer screen.

2.2 Woods' Immigration Game

Don Woods' *Immigration Game* has three colours, white cells (*empty, dead cells*) and two types of *live* cells, red, and blue (Wainwright, 1971, p. 14). The Immigration Game was intended to be a two-player game. Assume that the grid is initially all white squares. One player would turn some of the white squares into red squares and the second player would turn some of the white squares into blue squares. Then the two colours would compete against each other, blue versus red, until one of the colours is eliminated.

1. Survivals. When a cell survives with two or three live neighbours, its colour remains the same as it was before.
2. Deaths. When a cell with four or more neighbors dies, it becomes white. When a cell with one or zero neighbours dies, it becomes white.
3. Births. When an empty cell has exactly three live neighbours, the majority colour (red or blue) determines the colour of the newly born cell.

In our experiments, we follow the above three rules, but we do not have two players. As in Conway's Game of Life, we assume there is only one player. The computer screen is initially all white. The single player then switches some of the white cells to red or blue. The game runs for a specified number of steps, as it would with Conway's Game of Life. The single player in our experiments, for both games, is an evolutionary algorithm, not a human.

3. Evolutionary Algorithms

Evolutionary algorithms were inspired by Darwin's theory of biological evolution (Spears, 1998; Simon, 2013). An evolutionary algorithm includes operations for mutation and selection, applied to a data store, just as biological evolution applies mutation and selection to DNA. This combination (mutation, selection, and a data store) describes a minimal instance of an evolutionary algorithm. Optional extras include sexual recombination, segregation, genetic drift, coevolution, and symbiosis. We will use only mutation, selection, and a population of two-dimensional matrices for storing data.

3.1 Software: Golly and Python

In this article, we use the Golly Game of Life software, version 4.2 (Trevorrow and Rokicki, 2022), and the Python programming language, version 3.12. Golly was designed to be integrated with Python. Golly provides a flexible viewing environment for animating the growth of cellular automata on a computer screen. Python provides a programming environment that facilitates modifying and enhancing the capabilities of Golly.

The evolutionary algorithm code is implemented in Python (Turney, 2024). The population of digital organisms is represented as a list of two-dimensional Python matrices. Only one organism appears on the Golly screen at a time. A two-dimensional Python matrix is sampled from the list of matrices and written on the Golly screen, where its fitness is then evaluated. The screen is then cleared for the next organism.

The Game of Life is included in the Golly software. The Immigration Game is not included in Golly, but it is available in our GitHub repository in the file *Immigration.rule* (Turney, 2024).

3.2 Mutation, Selection, and an Evolving Population of Patterns

Our playing field consists of a 60×60 grid of squares, displayed in Golly. The grid is a toroid (a doughnut). Imagine a 60×60 grid of squares, then wrap the grid into a tube by joining the top of the grid to the bottom of the grid. Next, join the left side of the tube to the right side of the tube. This creates a finite toroidal grid of 60×60 squares, with no borders. The grid looks like a square plane in the Golly screen, but it behaves like a toroid. For example, if a pattern moves across the right border of the square plane, it will reappear at the left border. If a pattern moves across the top of the grid, it will reappear at the bottom of the grid.

The advantage of a toroidal grid over a square grid is that there can be artifacts at the edges of a square grid, which tend to disrupt the patterns in the grid. An unbounded grid would also avoid artifacts, but it could increase computation time.

The playing field (the Golly toroid) contains only one organism at a time. The population of organisms is stored in a list of matrices (in Python), one matrix for each organism. Each matrix in the list of matrices is initially filled with white (zero) and then colours are randomly added to the matrices (black for the Game of Life or red and blue for the Immigration Game). The matrices are 60×60, but an organism is initially limited to the 20×20 center of the matrix. We call this initial 20×20 configuration a *seed*.

In our experiments, we begin with a population of 1,000 randomly generated seeds. To evaluate the fitness of a seed, it is copied from the Python list and then written into the Golly toroid. As the seed grows in the toroid over time, it will tend cover the 60×60 grid. We allow the seed to grow for 100 steps, at which point it is then an *adult*.

To evaluate the fitness of an adult, we compare the adult with a *target*. The target is a 60×60 pattern that a human player creates, for any reason, aesthetic, scientific, or whimsical. We call this *breeding* rather than *evolution*, because breeding involves a future target or goal, whereas evolution is responding to the present circumstances.

We randomly sample two seeds from the population of 1,000 seeds. We score each of the two seeds. The seed with the higher score of the two is preserved in the population and the seed with the lower score is removed from the population. This reduces the population to 999. We then make a copy the seed with

the higher score and mutate the copy. The new seed is 20×20, so it has 400 squares. The probability of mutation is set to 0.05, so the expected number of mutations in the new seed is 20 (400×0.05). The mutated copy is the offspring of the winning seed. It might be more fit than the parent or it might be less fit. The new seed brings the population back up to 1,000.

To score a seed, we first allow it to grow to adult size, by running the game for 100 steps. We then compare the adult with the target. Each square in the adult (60×60 = 3,600 squares) is compared with the corresponding square in the target (also 3,600 squares). The method for scoring depends on whether we use the Game of Life or the Immigration Game. We repeat the scoring process 1,000,000 times.

In the Game of Life, one point is given for each case where both squares (the target square and the corresponding adult square) are black. Zero points are given if both squares are white. If one square is white and the other is black, one point is subtracted from the total score. In the initial population, the probability of a black square is 30% and the probability of a white square is 70%, but this ratio tends to change with selection, depending on the target.

In the Immigration Game, one point is given for each case where both squares (the target square and the corresponding adult square) are blue. One point is also given for each case where both squares are red. If one square is red and the other is blue, one point is subtracted from the total score. If a square is white, whatever colour the corresponding square has, a small penalty of 2% is applied. This is to encourage more red and blue matches. In the initial population, the probability of a red square is 15%, the probability of a blue square is 15%, and the probability of a white square is 70%, but this ratio changes with selection.

Note that the evolutionary algorithm cannot actually see the target pattern. It only receives a numerical fitness score that measures the similarity of the adult pattern to the target pattern. In effect, the evolutionary algorithm is blind.

A challenge here is that we want to make it easy for the human player to use the software, therefore we expect the human player to paint the desired pattern in broad strokes of colour, but the live squares in the game will tend to die when they are densely packed. The Game of Life and the Immigration Game do not tolerate high densities; they prefer lots of white space. This means that the fitness measure for the evolutionary algorithm should be tolerant of low densities (many white squares).

4. Breeding with Artificial Selection

With *natural selection*, nature determines how a population of organisms will vary over time. Successful organisms will pass their genes on to the next generation. Less successful organisms will die away. Natural selection does not plan for the future; it only responds to the present.

With selective breeding, *artificial selection* can be used to guide a population of organisms towards desired traits. Artificial selection, guided by humans, can plan for the future. New creatures can be bred for specific purposes, to serve future goals.

For human players of the Game of Life, the typical goal is to generate interesting images (Johnston and Greene, 2022). We use *selective breeding* as a tool that makes it easier for Game of Life enthusiasts to achieve their goals. The idea is that the user will paint broad swaths or strips of colour on the screen, defining a *target* for the evolutionary algorithm. The evolutionary algorithm will then evolve seed patterns that attempt to match the target.

It is not easy for the evolutionary algorithm to match the targets, because a Game of Life pattern requires many empty spaces around it, in order to survive, but the target swaths are wide strips of solid colour. In the Game of Life, a wide strip of solid colour quickly blows up chaotically.

The solution is to fill one screen with broad swaths of colour (the static target) and use a second screen for evolving Game of Life patterns (the growing organism). The swaths of colour on the first screen are compared with the sparse Game of Life patterns on the second screen. A Game of Life pattern gets one point for each case where the colour of a square in the first screen matches the colour of the corresponding square in the Game of Life pattern. When the colour on the first screen does not match the colour on the second screen, there is a small penalty. With evolution, the Game of Life pattern will try to match the swaths as well as it can, but it will not attempt to cover the whole swath, since too much crowding will cause cells to die.

4.1 Black and White: The Game of Life

In Golly, we use the rule *B3/S23:T60,60* to specify the Game of Life with a toroidal shape. *B3/S23* means a new life is born (*B*) when an empty square has three neighbours. A square survives (*S*) as long as it has either two or three living neighbours. *T60,60* specifies a toroid (*T*) that is 60 squares by 60 squares. Using a toroid instead of a finite plane allows us to avoid artifacts at the edges of the plane.

We will attempt to breed an organism that will form a specified desired shape. In Figure 1, our target shape is a thick vertical black stripe. We chose this stripe for its simplicity. Later we will introduce slightly more complicated target shapes, in Section 4.2.

We would like an evolved seed to grow, according to the rules of the Game of Life, until it looks like our target black stripe. However, the Game of Life prefers a density of about 20% to 30% black squares. A thick black stripe (100% black squares) is not stable, as we can see in Figure 2. The Game of Life needs lots of empty space to stay alive. The best that we can do is to approximate a solid black stripe by evolving a seed that will grow to an adult form that roughly sketches a black stripe, with many gaps and breaks.

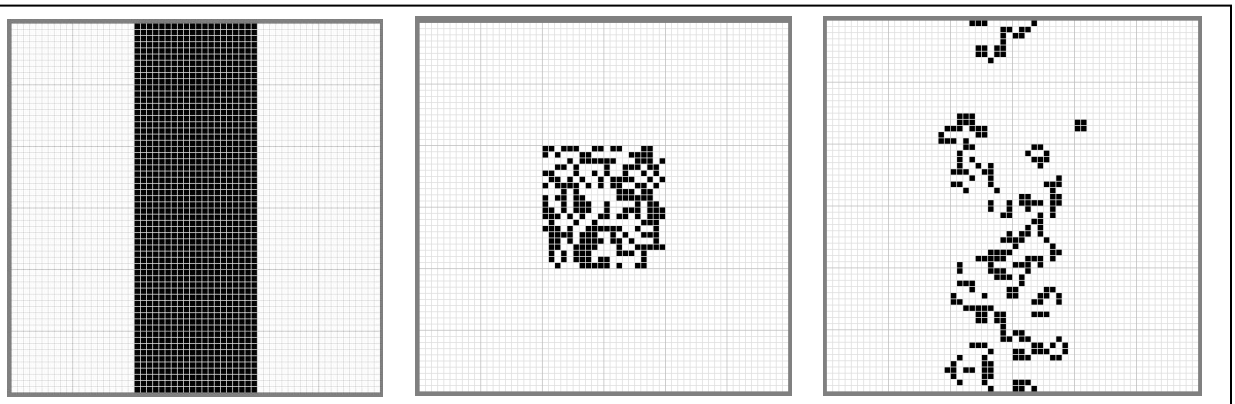


Figure 1: The adult form of the evolved seed is a rough approximation of the target black stripe, but the adult will gradually lose its shape as it ages.

In Figure 2, we can see how the target black stripe changes over time. We begin with a thick vertical black stripe (Step 0). We then allow the stripe to grow for 100 steps, according to the rules of the Game of Life. The target black stripe begins as a solid vertical bar, but then it quickly breaks up into many separate vertical bars. Thick strips of black are not stable in the Game of Life.

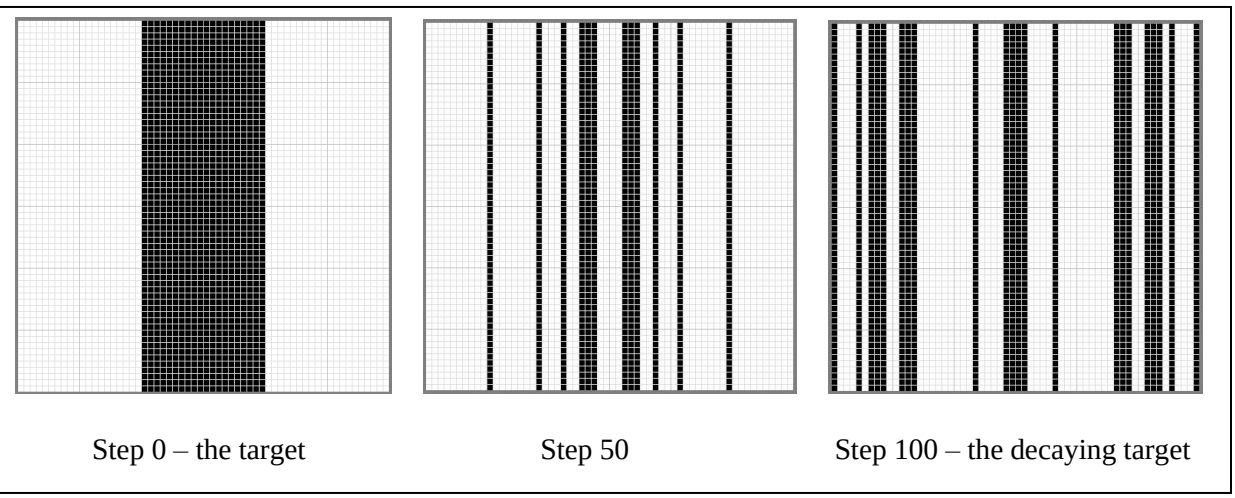


Figure 2: A dense band of black (Step 0) is not stable when using the rules of the Game of Life.

In Figure 3, we start with the seed and watch it grow into its adult form. The adult form of the seed is an attempt to create something like the target stripe in Figure 2. In the adult form in Figure 3, the majority of the black squares are in the middle third column of the toroid, as they are in the target black stripe. The

population of 1,000 seeds evolved 1,000,000 different patterns in an attempt to match the target stripe. The target black stripe is unstable (see Figure 2), but the adult form of the evolved seed is a rough approximation of the of the target black stripe (see Figure 3).

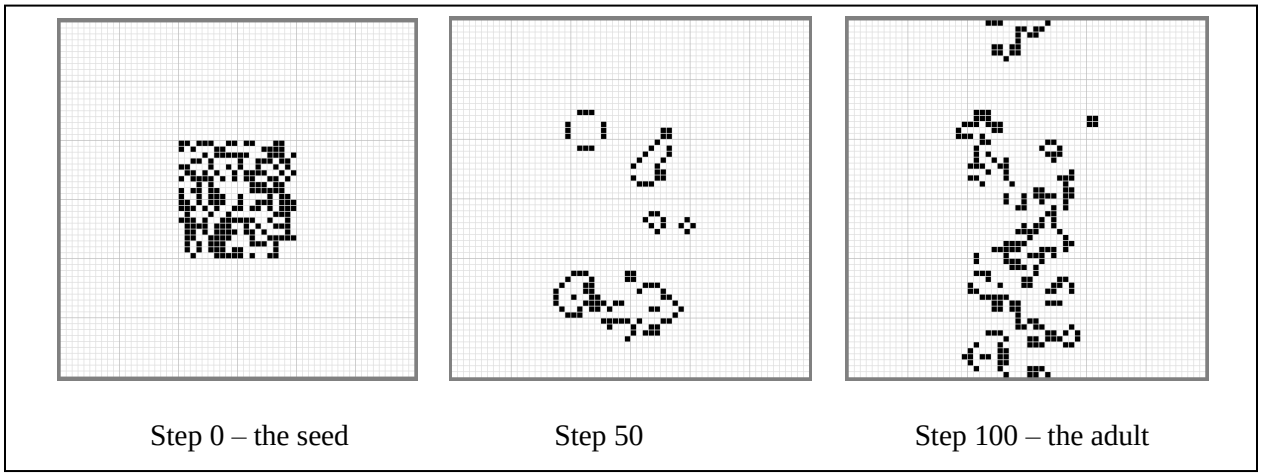


Figure 3: Over the course of a hundred steps in the Game of Life, the seed in Step 0 gradually grows to become the adult vertical stripe in Step 100.

In Figure 4, we see the adult form losing its cohesion over time. The connected shapes in Step 0 break apart and spread away from the middle vertical band in Steps 50 and 100. Note that the decaying adult (Figure 4) still shows a vertical concentration of black in the middle of the screen, whereas the decaying target has spread across the whole screen (Figure 2).

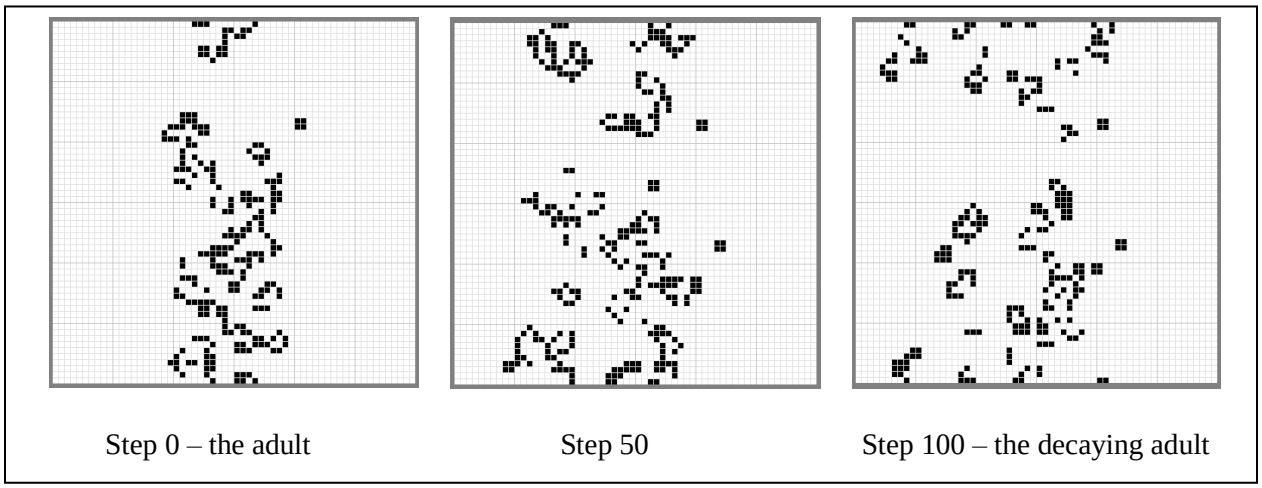
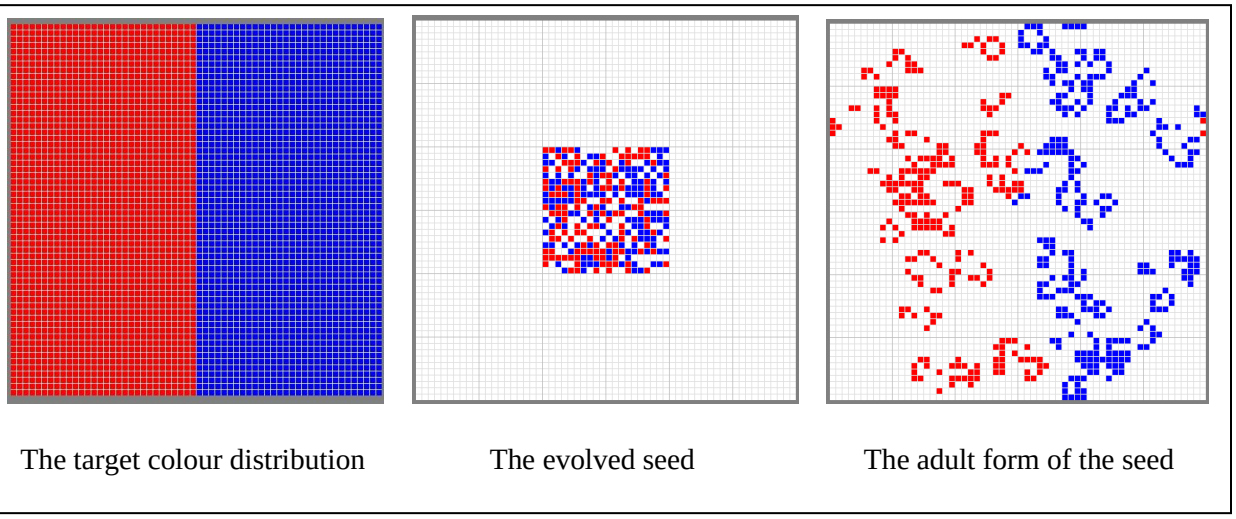


Figure 4: The adult form of the seed in Step 0 gradually decays away from a central vertical shape to a random scattering of objects.

4.2 Red, Blue, and White: The Immigration Game

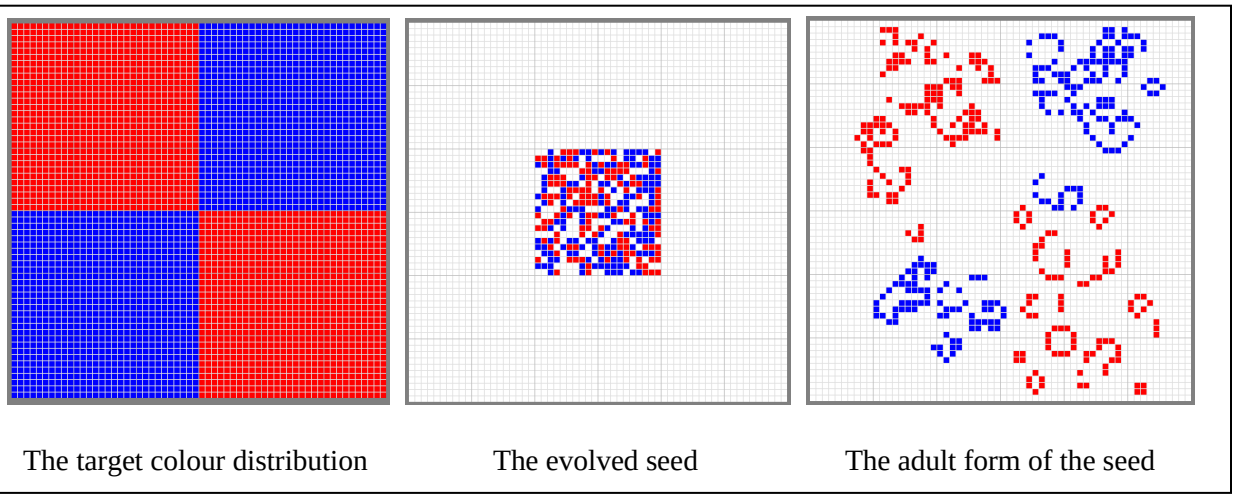
Now we move on from the Game of Life to the Immigration Game. With red and blue replacing black, the images become more interesting. In Figure 5, red and blue seem to be thoroughly mixed together in the evolved seed, yet the adult form has managed to split the colours apart. The evolved seed grows into two separate vertical bands of colour. This separation of the colours is impressive, but note that it required 1,000,000 births. Many seeds had to die in order to evolve one seed that was able to separate red from blue.



The target colour distribution | The evolved seed | The adult form of the seed

Figure 5: With 1,000,000 births, it is possible to separate a thorough mixture of red and blue (the seed).

Figure 6 has a slightly more complex target than Figure 5. There are four separate regions, yet the seed is able to assign the right colour to each region.



The target colour distribution | The evolved seed | The adult form of the seed

Figure 6: Four separate regions are coloured according to the target.

The following target has three separate red regions and a blue stripe with a ninety-degree angle in it. Nonetheless, the evolutionary algorithm is able to approximate the shape of the target.

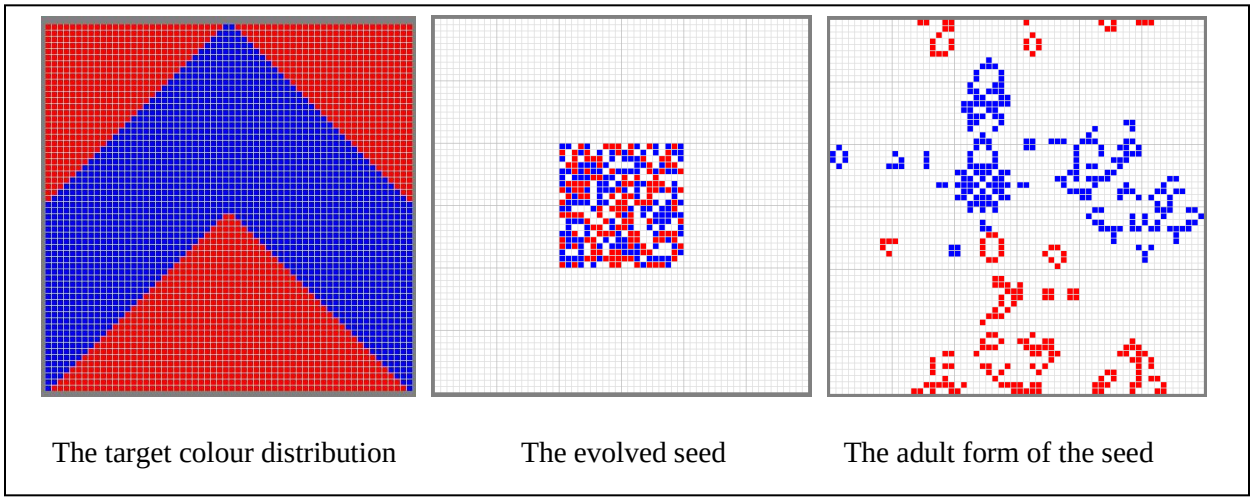


Figure 7: Four separate regions of colour with diagonal lines.

The final target, Figure 8, is a repeat of the previous target, Figure 7. Although the two targets are the same, the two adult forms are significantly different. This is true for all of the seeds and adults. It is very unlikely that two identical targets will result in two identical seeds or two identical adults. Given that 1,000,000 adult patterns are generated in a run, it is quite unlikely that two adults will look the same.

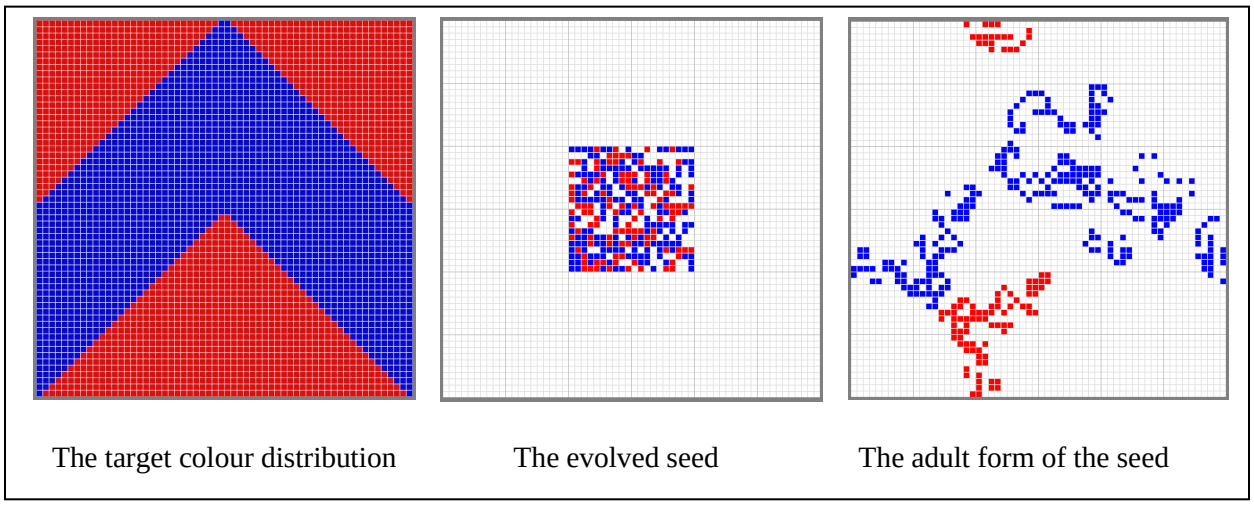


Figure 8: This is the same target as the target in Figure 7, but the adults are notably different.

5. Related Work

In past work (Turney, 2020), we simulated Game of Life organisms that could gradually increase their fitness over many generations, by adding layers of different types of reproduction. The first layer was simple asexual reproduction. The second layer added a more complex form of asexual reproduction. The third layer added sexual reproduction. The fourth layer added symbiosis. The fitness of the organisms was defined simply as how large they grew within a fixed time limit (a fixed number of steps in the game).

In the current article, the reproduction of Life organisms is based on simple asexual reproduction. It is different from the earlier work (Turney, 2020) in that the evolution of the organism is guided by various *targets*. A target is used to calculate the fitness of an organism. Organisms that match well with the target are more likely to reproduce (with mutations) and organisms that do not match well tend to die (they leave the population). The human user of the software acts as a breeder, by generating the targets that guide the evolution. This is unlike the past work (Turney, 2020), where growth was the only goal.

Perhaps the most similar work to ours is *Evolving Interesting Initial Conditions for Cellular Automata of the Game of Life Type* (Alfonseca and Soler Gil, 2012). We are both exploring the Game of Life and we are both particularly interested in the initial conditions. The initial conditions determine the future results. In our case, the evolutionary algorithm generates an evolved seed by randomly creating 1,000,000 seeds, using mutation and selection, until it finds the seed that best approximates the target. In the case of Alfonseca and Soler Gil, they explore time-dependent rules and alternating rules. Both of us use a 60×60 toroid.

Alfonseca and Soler Gil wrote that "… each complete execution of the genetic algorithm takes over half an hour, because the CA used by the algorithm must be run for each set of initial conditions in the population through generations 1 to 54 to compute their fitness…" (page 60). This is quite similar to our approach. The main difference is that our *targets* are arbitrary bands of colour, whereas their interest is in common objects in the Game of Life, such as gliders, R-pentominoes, and exploders.

6. Future Work

In this article, the experiments focus on one organism at a time in the toroid. In future work, we plan to increase the size of the toroid or use an unbounded plane. This will allow us to scatter many seeds on the playing field, either randomly or systematically. For example, several copies of the seed in Figure 1 could be arranged in a large hollow square shape, with the left and right sides of the square sketched out by the seeds oriented as in Figure 1, and the top and bottom sides of the square sketched out by the seeds turned ninety degrees. As the seeds grow, they will stretch out and join, making a hollow square.

Here we have focused on evolving a static target pattern, but dynamic patterns are also possible. For example, the fitness score could be based on a series of target patterns arranged in time. After many generations of evolution, a seed would evolve that is able to dynamically alter its shape, conforming to the series given in training, due to mutation and selection. This would create a kind of animated movie, constructed from evolved Game of Life structures.

Another direction for future work would be to increase the colour palette beyond three colours (red, white, and blue). We have done some preliminary experiments with this, but it is challenging to balance the growth of the colours. A colour that grows quickly can erase colours that grow slowly. This problem might be addressed by exploring other cellular automata rules, beyond the Game of Life rules.

The big picture is that selective breeding of automata has the potential to create new kinds of structures, significantly different from the careful, technical, precise structures that are created in the Game of Life community (Johnston and Greene, 2022). Breeding lacks precision, but it can be quite creative.

7. Conclusions

Johnston and Greene (2022) provide an excellent survey of research in the Game of Life. They describe early discoveries in Life, based on *random fumbling* (their term, not ours), followed by classifying structures into various basic types, then combining parts to make complex interactive wholes. Their book spans twelve chapters, more than four hundred pages of discoveries about Life. Since Life is known to be a universal computer, it seems that there should be no limit to the growth of our knowledge about Life. This is supported by the progress we have seen over more than 50 years of Life (Gardner, 1970).

Johnston and Greene (2022) describe various software tools that have been constructed to support constructions in Life, but evolutionary algorithms are not discussed. The word *evolution* occurs 58 times in their book, but only in the generic sense of *change*, not in the sense of *mutation and selection* (Spears, 1998; Simon, 2013). The field of Life has evolved by a kind of *manual* mutation and selection of Life structures, but there is little work on *computational* mutation and selection of Life structures. One explanation for this is that the Game of Life community is essentially an *engineering* community, interested in building things by step-by-step design, whereas evolutionary algorithms build structures by mutation and selection, which is more biology than engineering.

We believe that *breeding* may interest the Game of Life community more than *natural selection*, because breeding is driven by future goals, whereas natural selection is driven by current circumstances. The Life community is driven by goals. The challenge is how to breed Life structures. Evolution is easiest when small mutations to genes result in small changes to subsequent adults. It is the nature of the Game of Life that small changes to Life patterns can cause large changes to the adult forms. The idea of using a *target* to guide evolution provides a method for evolving new structures by selective breeding.

Acknowledgments

Thanks to Andrew Trevorrow, Tom Rokicki, Tim Hutton, Dave Greene, Jason Summers, Maks Verver, Robert Munafo, Brenton Bostick, and Chris Rowett, for developing the Golly cellular automata software. Thanks to Saif Mohammad, David Nadeau, and Nate Gaylinn, for helpful comments and suggestions.

References

Alfonseca, M., and Soler Gil, F.J. (2012). Evolving interesting initial conditions for cellular automata of the Game of Life type, *Complex Systems*, *21* (1), 57-70. https://doi.org/10.25088/ComplexSystems.21.1.57

Berlekamp, E.R., Conway, J.H., and Guy, R.K. (1982). *Winning Ways for Your Mathematical Plays*, Academic Press.

Gardner, M. (1970). Mathematical games: The fantastic combinations of John Conway's new solitaire game 'Life'. *Scientific American*, *223* (4), 120-123. https://doi.org/10.1038/scientificamerican1070-120

Johnston, N., and Greene, D. (2022). *Conway's Game of Life: Mathematics and Construction*. https://conwaylife.com/book/conway_life_book.pdf

Poundstone, W. (2013). *The Recursive Universe: Cosmic Complexity and the Limits of Scientific Knowledge*, Dover Publications.

Simon, D. (2013). *Evolutionary Optimization Algorithms: Biologically Inspired and Population-Based Approaches to Computer Intelligence*, Wiley.

Spears, W.M. (1998). *Evolutionary Algorithms: The Role of Mutation and Recombination*, Springer.

Trevorrow, A., and Rokicki, T. (2022). *Golly Game of Life Software*. https://golly.sourceforge.io/

Turney, P.D. (2020). Symbiosis promotes fitness improvements in the Game of Life, Artificial Life, MIT Press. *26* (3): 338-365. https://doi.org/10.1162/artl_a_00326

Turney, P.D., (2024). *Breeding Monochrome Life*. https://github.com/pdturney/breeding-monochrome-life

Turney, P.D., (2024). *Breeding Colourised Life*. https://github.com/pdturney/breeding-colourised-life

Wainwright, R.T. (1971). *Lifeline Volume 2*. Retrieved April 25, 2024, from LifeWiki. https://www.conwaylife.com/wiki/Lifeline_Volume_2