Optimization of Initial Conditions for Complex Behavior in the Game of Life Using a Genetic Algorithm

Nicholas DeGroote, Lynn Pickering, Vita Borovyk, and Owen Traubert
College of Engineering and Applied Sciences
University of Cincinnati

Abstract

The initial conditions of the Game of Life Cellular Automata are trained by a multi-population Genetic Algorithm with the goal of achieving complex behavior. The multi-population Genetic Algorithm ensures that genetic diversity is maintained while improving the performance of the GA as a whole. Complex behavior is quantified for use by the GA's fitness function using three metrics: fractal fitness, cluster distance fitness, and cluster size fitness. Fractal fitness measures the fractal dimension. Cluster distance fitness measures the number of cells clustered and their distance from the center of the board, and cluster size fitness measures the size of the larges cluster on the board. The GA finds initial conditions that result in simulations that achieve high fitness values in accordance with each fitness function.

1 Introduction

Cellular automata (CA) are a collection of discrete cells which form grid, and can take on a number of states [1]. The grid is configured in a way such that the next state of each cell is determined by the current state of the cells near it, called neighbors. The states are determined by a set of rules, which determine what each cell what it will do next based on the configuration of its neighbors. States are often binary, representing things such as on or off, although more than two states can be incorporated. Phenomena resembling CA can often be seen in many real-world systems such as patterns on mollusk shells, growth of crystals, or the movement of fluids [2].

The evolution of the CA through time is highly dependent on its initial conditions. Vastly different behavior can be observed with varying initial conditions despite the rule set remaining constant. Differing initial conditions can produce behavior ranging from trivial solutions with no activity to complex emergent behavior.

A cellular automation called the Game of Life (GoL) is the focus of this research. The GoL was originally developed by John Horton Conway in 1970 and has the potential to produce complex patterns and emergent behavior [3]. In the GoL, cells can take on one of two states: *alive* or *dead*. The concept for the GoL features only four rules. First, any alive cell with fewer than 2 neighbors dies. Second, any alive cell with exactly two or three neighbors survives. Third, any alive cell with more than three neighbors dies. Finally, any dead cell with exactly three neighbors becomes alive.

This research seeks to develop initial conditions for the GoL which evolve the most complex and interesting behavior over time. Complexity is quantified through a variety of different metrics so the

results of the CA simulations can be compared objectively. A genetic algorithm (GA) is used to find the initial conditions which produce the most complex results. A variety of different parameters are tested and the results analyzed to see which attributes have the greatest effect on complexity.

2 Background

Advances in computing power in recent years have made the application of gradient-free heuristic search methods to complex systems, and the development of desired emergent behavior increasingly practical. These methods have shown to produce near-optimal results in a fraction of the required computation time for exact methods. They provide a way to guide a search through very large solution spaces for a global optimum. A GA is one such type of heuristic algorithm which searches for a global optimum by imitating the process of biological evolution.

A GA has a collection, or *population*, of *chromosomes*, where chromosomes represents a potential solution to the optimization problem. Each chromosome in the population has a number of *genes*, where each gene represents a component of the solution. The quality, or *fitness*, of each chromosome is determined by a fitness function. The GA runs for a specified number of generations, with the fittest chromosomes in the population selected as *parents* to pass on their genes to *children*. Each child chromosome is composed of a combination of genes from both of its parents.

GAs posses a variety of desirable characteristics which make them a good choice as an optimizer for this research. They are a black-box optimization technique, and therefore do not impose any constraints for fitness evaluation, enabling them to be used in complex simulation environments [4]. They are also adaptable to different solution concepts, which can help to design a desired type of emergent behavior.

One of the major challenges associated with determining the complexity of a CA is finding a quantifiable metric for complexity. Previous studies have used metrics such as counting the number of "interesting" structures found during a CA simulation [5]. This approach is of course quantifiable, but still lacks a general description of what makes the observed structures interesting. As such, a metric is desired that allows complexity to be evaluated at a more abstract level.

3 Methodology

A concept called *multi-population* is used in this project, where a GA uses multiple populations which only occasionally communicate with one another. Studies on multi-population have shown it to be an effective method for improving the performance of evolutionary algorithms such as GAs [6] [7]. The goal of using multiple populations is to ensure that genetic diversity is maintained by allowing different populations to explore different regions of the solution space. Multi-population also makes it less likely that chromosomes with useful genetic information but low fitness will die out prematurely.

The developed multi-population GA lets each chromosome perform crossover with only its own population; however, the crossover and mutation operators function the same as for a single-population GA. At the end of each generation, there is a chance for a *migration* event to take place. When a migration occurs, two populations are randomly selected, and the best chromosome from one of the populations is copied to the other. This operation exists to allow communication between the populations and occasionally introduce even more genetic diversity. At the end of each

generation for the GA, the size of each population is updated based on the average fitness of each population. Fitter populations have a higher chance of increasing in size, while less fit populations have a higher chance of decreasing in size. The goal of these two mechanics is to balance the protection of genetic diversity by allowing certain low-fitness chromosomes to survive while also incentivizing good performance by rewarding the most successful populations.

The initial conditions for the CA developed in this research form an N x N grid, which can be reshaped to be represented by a binary string of length N^2, where N is the length of a side of the grid. A value of zero in the string represents a cell which is dead at timestep zero, while a value of one represents a cell which is alive at timestep zero. The CA simulation is run in an environment of cells with dimension 2N x 2N, having the initial conditions centered on this board. For example, a 32 x 32 initial condition set is centered on a 64 x 64 board. The fitness of the CA is evaluated at iterations 40 to 60 to allow long enough for the simulation to stabilize [5].

The fitness functions for the GA attempt to maximize various metrics of complexity and interesting behavior. This section describes the different complexity measures used in this research: fractal complexity, cluster distance complexity, and cluster size complexity. Some background information on the complexity measures is included in this section to better illustrate the rationale for its implementation.

3.1 Fractal Fitness

The first fitness function used to measure complexity is fractal fitness. Previous research examined the relationship between fractal dimension and complexity, beauty, subjective visual interest, and found correlation. Corbit and Garbary examined the shapes of fronds of brown algae and found high correlation between maturity level and fractal dimension as well as a high correlation between subjective ratings of complexity and fractal dimension as measured by box counting [8]. Forsythe et al. found that fractal dimension as measured by box counting was a predictor of subjective rankings of beauty for images in general, but was most suited to evaluate abstract and natural beauty. Natural images, they found, had the highest fractal dimension [9].

Fractal dimension is also a correlate of autopoiesis, an autopoeitic system being a network of component-producing processes whose interactions generate the very same network of processes that produced them while remaining a distinct entity [10]. All living cells and organisms, ecosystems, and many other interesting complex systems are autopoietic. Beer rigorously proved that self-perpetuating GoL entities such as blocks, blinkers, and gliders are autopoietic, and have higher fractal dimension than random patterns [10]. We assert that the rapidly growing and changing patterns seen at the beginning of a GoL run, which, as can be seen in Figure 1, have a higher fractal dimension than quiescent patterns at the end, are loosely autopoietic because they are networks of interdependent local processes that perpetuate themselves despite not being strictly closed and bounded. This resembles the difference in real-world natural systems between cells of an animal and the spores of a coral reef.

For fractal fitness, box counting is also used. Fractal fitness is calculated at a single timestep in the CA simulation by normalizing a deviation from a desired range for fractal dimension. Fitness for a chromosome F, is calculated as the mean fitness at all timesteps evaluated in its GoL simulation:

$$F = \frac{1}{\#steps}\sum^{i} F_i \quad (1) \qquad F_i = e^{-4\Delta_{D_i}} \quad (2)$$

$$D_i = \frac{1}{3} log_2(S_{1/8}) \quad (3) \qquad \Delta_{D_i} = \begin{cases} 1.58 - D_i & D_i < 1.58 \\ 0 & 1.58 < D_i < 1.9 \\ D_i - 1.9 & D_i > 1.9 \end{cases} \quad (4)$$

$$S_{1/8} = \frac{N_{1/8}(Z_i)}{N_1(Z_i)} \quad (5)$$

Where F_i is the fractal fitness at one timestep, D_i is the calculated fractal dimension at one timestep and Δ_{D_i} is the deviation from our ideal fractal dimension range at that timestep, and $N_j(Z_i)$ is the number of boxes it takes to cover the pattern of live cells Z at timestep i, Z_i, on a GoL board given boxes with side length j units. The unit size for our purposes is defined as 8 GoL cells, which was chosen arbitrarily from possible reasonable values of 8 and 16 given our board size of 32 by 32.

The parameters 1.58 and 1.9 for our desired range in Eq. 4 were chosen based on existing interesting fractals with constraint due to two dimensions: interesting natural processes such as random walks, diffusion-limited aggregation clusters, and 2D percolation clusters have fractal dimensions that are very near or within this range [11]. The precise range isn't of crucial importance as we determined, as can be seen in Figure 1, that GoL rules and random initial conditions resulted in low fractal dimension that dropped off asymptotically throughout evolution of initial conditions, and therefore any fractal fitness value above 0.01 is an improvement. Additionally, very high fractal dimension isn't compatible with the standard GoL rules as overly high density will result in cells dying.

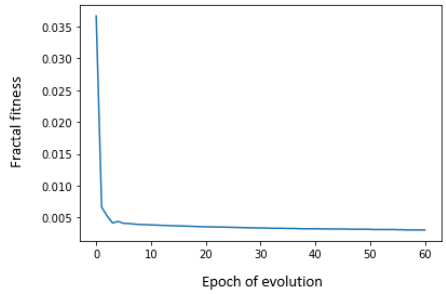


Figure 1: Mean fractal fitness of 10,000 runs of GoL with random initial conditions

3.2 Cluster Distance

The second fitness function examined was cluster distance. This function determines the mean center of a cluster of cells, where a cluster is defined as a contiguous group of live cells, and any neighbor in a cell's Moore neighborhood counting as a contiguous neighbor. Based on this principle, the distance from the center of the board for all clusters is computed. Fitness for a chromosome was simply defined as the sum distances between the centers of all clusters of at least size k and the center of the board, averaged over all time steps considered for fitness. The fitness, F, is calculated as the mean fitness at all evaluated timesteps in GoL evolution:

$$cardinality(c_j) \geq k \tag{6}$$

$$F = \frac{1}{\#steps}\sum^{i} F_i \tag{7}$$

$$F_i = \sum^{j} d(c_j, center) \tag{8}$$

$$c_j = \frac{1}{k}\sum_{1}^{k}(x_j^k, y_j^k) \tag{9}$$

where $d(a,b)$ is the distance between two points and (x_j^k, y_j^k) is the k$^{\text{th}}$ member of cluster j.

Figure 7, shown later, depicts GoL rules run on random initial conditions, shows outward growth is not an inherent property of the rules of GoL. This outward growth is interesting because it encourages latent growth-forming patterns to arise and because it mimics natural outward growth.

Circle formation for simple agents, as studied by Tanaka and Swaminathan and Minai, also plays a role in the design of the cluster distance fitness algorithm [12] [13]. The use of clusters for quantification turns each cluster into a sort of agent whose goal is to be as far away from the center of the GoL board as possible. This property combined with the property of GoL cells requiring a certain density to live, i.e. distance from their neighbors, may lead to circle formation.

3.3 Cluster Size

The third fitness function examined is cluster size. This function calculates fitness based on size of the largest clusters found at each timestep. Larger clusters have a higher probability of percolating through the entire system and being spanning cluster, therefore increasing the probability that any given live cell is a member of the largest cluster [14]. Activity and change in GoL occurs in clusters, with the most vibrant activity tending to happen towards the beginning of the evolution in larger clusters. Given the hypothesis that these clusters are loosely autopoietic, as well as the behavior of systems at the threshold for percolation, we found this to be an interesting to parameter to optimize. Fitness for a chromosome was simply defined as the largest cluster size at all time steps considered for fitness. Fitness, F is calculated using:

$$F = \frac{1}{\#steps}\sum^{i} F_i \tag{10}$$

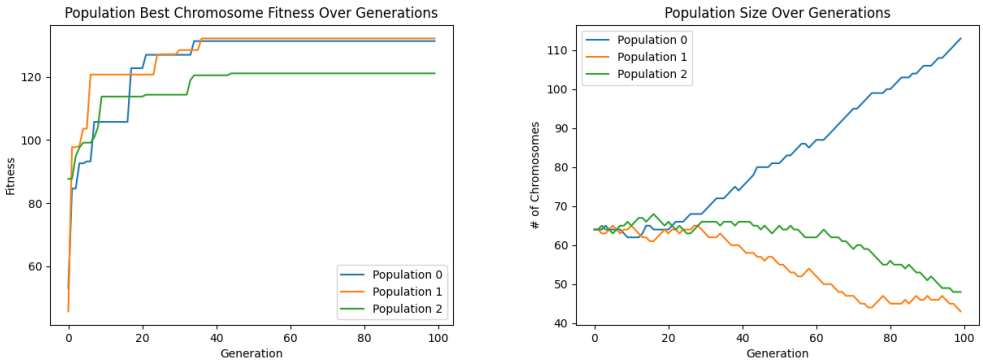
$$F_i = \sum_{y=0}^{y_{max}}\sum_{x=0}^{x_{max}} v(y,x) \tag{11}$$

$$v(y,x) = \begin{cases} (y,x) & (y,x) \in C \\ 0 & (y,x) \notin C \end{cases} \tag{12}$$

and C is the cluster with the most members.

4 Results

For all three implemented fitness functions, the GA was able to converge to a final value. Figure 2a shows an example of a typical convergence plot for the cluster distance fitness function with the fittest chromosome from each population at each generation. From this type of plot, it was determined that the GA would run for 100 generations because this provides enough time for the GA to converge. Figure 2b shows the size of each population over the run of the GA.



(a) Best chromosome fitness vs. generations.

(b) Population size vs. generations.

Figure 2: An example of a typical convergence plot for the GA as well as the individual population sizes. Cluster distance was used as the fitness function.

One of the goals of this work was to observe the interaction of the rules of GoL with each of the fitness functions. Some of the configurations that would clearly be optimal on the entire space of $\{0,1\}$-vectors on the square lattice are incompatible with the rules. For example, a full board of alive cells maximizes the Cluster Size function, but is not possible to evolve to by following GoL rules. In subsections 4.1, 4.2, and 4.3 the structure of the solutions are discussed that the GA was able to find for each of the fitness functions at their evolved state, during time steps when the fitness values were measured.

4.1 Fractal Fitness

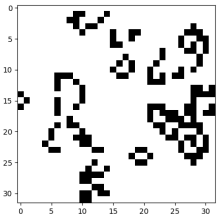
The GA was configured to maximize the values of fitness functions over timesteps 40 to 60 in the CA evolution. Unfortunately, this turned out to be an unsuccessful choice for the fractal dimension function on 32 by 32 board. The GA solutions stabilized to a fixed point configuration by approximately the 30^{th} timestep. Therefore, on the given interval of interest, the evolution appears as a still picture of one or two simple stable shapes as shown in Figure 3.

This was a surprising result as outcomes found by the other two complexity functions, discussed in the following subsections, have higher fractal dimension in the interval $[40,60]$ than the configu-

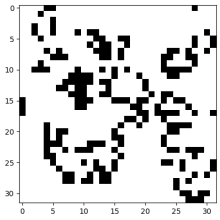
Figure 3: Typical fractal fitness configurations after 60 timesteps

rations found by the fractal dimension function itself. A possible explanation is that good solutions for fractal dimension form a larger set and are more sparse in the search space compared to those the other two functions, and more difficult for the GA to find reasonable options.

In an attempt to improve the outcomes for fractal fitness, we introduced a modified fractal fitness function as a product of the initial fractal dimension function and the density of the alive cells, that maximizes both as a result. This approach produced final configurations, shown in Figure 4, which did not stabilize as early. It is also interesting to note that fractal fitness produces the observed quiescent, static patterns starting almost exactly on the 40th epoch, as can be seen in Figure 5. Also evident from Figure 5, initial conditions evolved with fractal fitness had very low modified fractal fitness and relatively high fractal fitness, while initial conditions evolved with modified fractal fitness have much higher modified fractal fitness and relatively low but still appreciable fractal fitness.



(a) 40 timesteps



(b) 60 timesteps

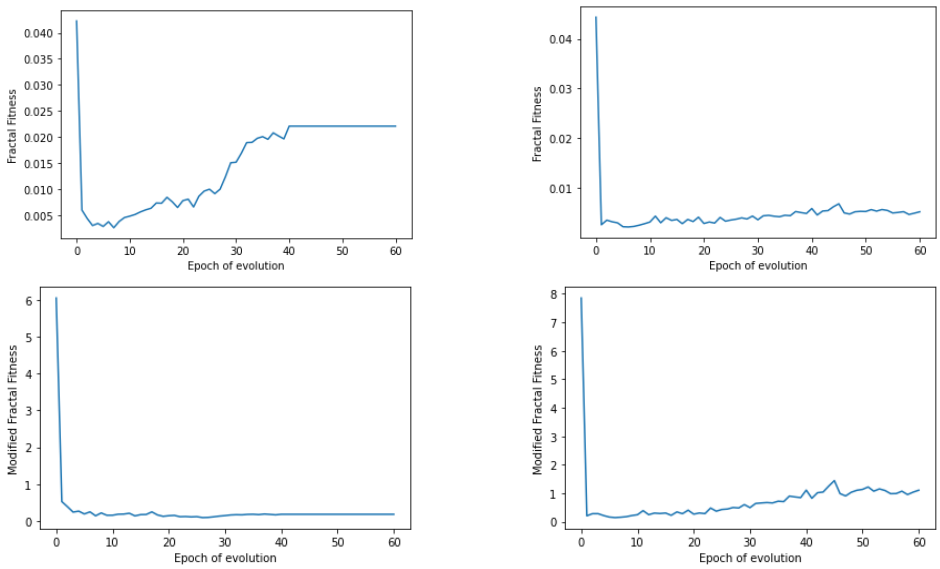
Figure 4: Modified fractal fitness (fractal dimension times density) solution after 40 and 60 timesteps

4.2 Cluster Distance

An optimal configuration for the cluster distance function would be a board filled with single cell clusters, spread out uniformly and having exactly one dead space between nearest neighbors. This, however, is incompatible with the GoL rules, as the smallest stable cluster for its evolution is the three-cell cluster in a shape of a corner. Hoping to see more interesting structures, it was decided to further bound the size of what is considered a cluster from below, choosing the default value to be 6. Figure 6 has several typical outcomes for such a setup. Many clusters of the minimal allowed size are present and they are distributed towards the edge of the board, both of which are expected.

However, the leaf clusters in these configurations do not remain stable. As evolution progresses, they interact with the disorganized part of the alive group and dissipate. Therefore these formations appear temporarily, at around the time range when we were performing measurements.

When the runs of numerous chromosomes are averaged together, it becomes clear that cluster distance had great success producing circle-like shapes. Figure 7 shows the results of 36 cluster distance chromosomes producing a hollow oval with heavy corners, a near inversion of the distribution from random initial conditions.



(a) Fractal fitness chromosomes

(b) Modified fractal fitness chromosomes

Figure 5: Fractal fitness and modified fractal fitness ratings for two groups of chromosomes: one evolved with fractal fitness and the other evolved with modified fractal fitness

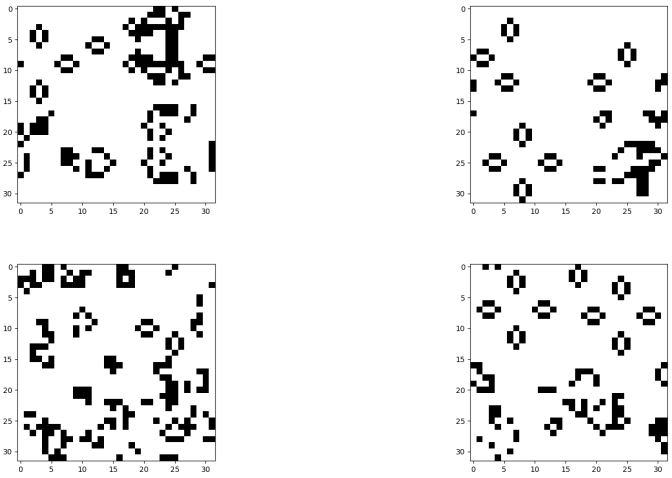


Figure 6: Various typical Cluster Distance configurations after 60 timesteps

4.3 Cluster Size

The optimal configuration for the cluster size function is one that contains a connected component that spans the entire lattice. While this is possible to achieve under the GoL evolution, it is difficult as the natural tendency for GoL evolution is to spread living cells around and break large clusters apart. Because our measurements are made at relatively early stages, the GA was able to find good solutions. Figure 8 demonstrates the behavior of a typical optimal solution. In the early time steps

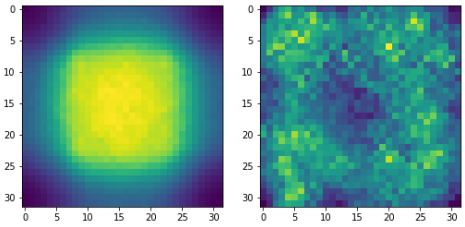
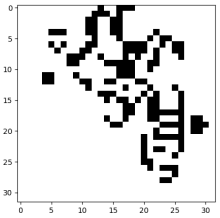
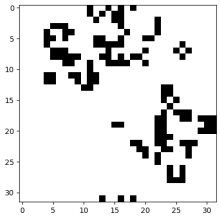


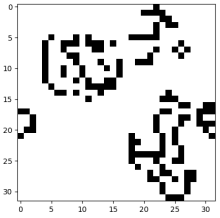
Figure 7: Mean distribution of cells in 10,000 independent evolutions of GoL with random initial conditions vs 36 initial conditions chromosomes evolved with cluster distance



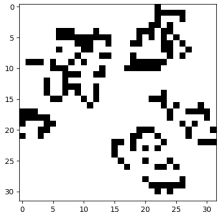
(a) 43 timesteps



(b) 48 timesteps



(c) 55 timesteps



(d) 60 timesteps

Figure 8: Typical Cluster Size configuration at various timesteps within measurement interval

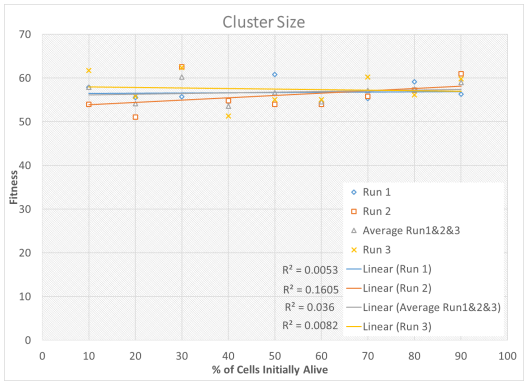
of the evolution, this solution remains fully connected, then breaks apart into separate pieces, which grew independently. These separate pieces then come back together to form one large cluster exactly by the time we start measuring fitness at time step 40. During the measurement interval, towards time step 60, the large cluster slowly becomes disconnected.

Cluster size optimized the size of the largest cluster, doubling the mean size of the largest cluster found at each time step from 23.7 cells using random initial conditions to 45.3 cells using chromosomes evolved with cluster distance. Interestingly, optimization with cluster distance had little effect on the mean size of clusters. Mean cluster size for random initial conditions is 8.1 cells, while it was 10.1 for our chromosomes evolved with cluster size fitness.

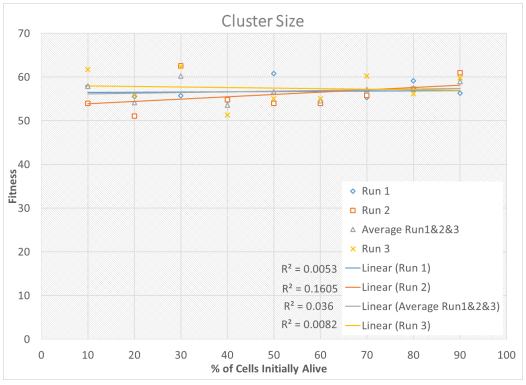
4.4 Density Analysis

An analysis was performed to determine the effects of the initialized density of the GA chromosomes for each fitness function. The percentage of cells alive in the initialized chromosomes were set from 10% up to 90% in 10% increments, for each fitness function. The board size tested was

(a) Cluster Distance Density Analysis



(b) Cluster Size Density Analysis



(c) Cluster Size Density Analysis

Figure 9: Density Analyses for cluster distance and cluster size

32 by 32. Each fitness measure was run three times to ensure that results were relevant. Figure 9a shows the density analysis for fractal fitness. This fitness measure had the least correlation to the density of the initialized chromosomes, there is none at all. Figure 9b and 9c show the results for the density analysis of cluster distance and cluster size, respectively. Cluster size produced similar values regardless of run and initial density. Cluster distance, while also producing similar fitness values at each density, was positively correlated to increasing density. The R^2 value for the line fit to the average of the three runs for cluster distance was 0.7031.

5 Conclusion

A multi-population GA has been developed and applied to develop complex behavior in the GoL Cellular Automata. The GA optimized the initial board of the GoL simulation in accordance with three fitness functions: Fractal fitness, Cluster Distance fitness, and Cluster Size fitness. It has been shown that the GA finds interesting initial conditions in accordance with each fitness function, and these solutions and outcomes have been discussed and compared.

One of the most challenging aspects of this project was dealing with time constraints and limited computing resources to run a wider net of simulations. In the future, given more time, the authors would run simulations on larger board sizes, especially for fractal fitness because it achieved higher

fitness values with larger boards. As the GoL rules were used for these simulations, future work would include variations on the GoL rules for the simulations. Another future topic that is interesting finding modularity in the initial conditions that lead to interesting results. To achieve this, the fitness function would alternate between two functions over the simulation, inspired by [15].

References

[1] D. Waters, "Von neumann's theory of self-reproducing automata: A useful framework for biosemiotics?," *Biosemiotics*, vol. 5, 04 2012.

[2] D. Green, "Cellular automata models in biology," *Mathematical and Computer Modelling*, vol. 13, no. 6, pp. 69–74, 1990.

[3] M. Gardner, "Mathematical games," *Scientific American*, vol. 223, no. 4, pp. 120–123, 1970.

[4] D. Icl, Anzan, R. Lung, and A. Andreica, "Evolutionary computing in the study of complex systems," vol. 1, 01 2011.

[5] M. Alfonseca and F. Gil, "Evolving interesting initial conditions for cellular automata of the game of life type," *Complex Systems*, vol. 21, pp. 57–70, 03 2012.

[6] A. C. Jihane, A. El Imrani, and A. Bouroumi, "A multipopulation cultural algorithm using fuzzy clustering," *Appl. Soft Comput.*, vol. 7, pp. 506–519, 03 2007.

[7] P. Siarry, A. Âtrowski, and M. Bessaou, "A multipopulation genetic algorithm aimed at multimodal optimization," *Advances in Engineering Software - AES*, vol. 33, 04 2002.

[8] J. D. Corbit and D. J. Garbary, "Fractal dimension as a quantitative measure of complexity in plant development," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 262, pp. 1–6, 10 1995.

[9] A. Forsythe, M. Nadal, N. Sheehy, C. J. Cela-Conde, and M. Sawey, "Predicting beauty: Fractal dimension and visual complexity in art," *British Journal of Psychology*, vol. 102, pp. 49–70, 01 2011.

[10] R. D. Beer, "Characterizing Autopoiesis in the Game of Life," *Artificial Life*, vol. 21, pp. 1–19, 02 2015.

[11] M. Sahini, *Applications of Percolation Theory*. London: CRC Press, 1994.

[12] O. Tanaka, "Forming a circle by distributed anonymous mobile robots," bachelor thesis, Department of Electrical Engineering, Hiroshima University, Hiroshima, Japan, 1992.

[13] S. Karthikeyan and A. A. Minai, *A General Approach to Swarm Coordination using Circle Formation*, pp. 65–84. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[14] K. Christensen, "Percolation theory." MIT Lecture Notes Collection, 09 2002.

[15] N. Kashtan and U. Alon, "Spontaneous evolution of modularity and network motifs," *Proceedings of the National Academy of Sciences*, vol. 102, no. 39, pp. 13773–13778, 2005.

6 Appendix

Example videos of the CA simulations found by the GA:

1. Cluster Distance, 32 x 32: `https://youtu.be/Df4FYyycG9s`
2. Cluster Distance, 64 x 64: `https://youtu.be/U8oY0Yk_D_g`
3. Fractal 32 x 32: `https://youtu.be/WZTcs-xY06E`
4. Cluster Size: 32 x 32: `https://youtu.be/RAMV8SjO888`
5. Fractal, 128 x 128: `https://youtu.be/tXBnsQ-C_YI`

Project GitHub repository:
`https://github.com/NickDeGroote/Complex_Systems_SS21`