



Implementation and Evaluation of an Order Parameter for the Reaction-Diffusion Model in a Cellular Automaton

How does an order parameter perform on a Reaction-Diffusion model implemented in a Cellular Automata?

Daniel Puente Barajas¹

Supervisor(s): Dr. M. Skrodzki¹, Dr. A.B.T. Barbaro¹

¹EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023


Name of the student: Daniel Puente Barajas

Final project course: CSE3000 Research Project

Thesis committee: Dr. M. Skrodzki, Dr. A.B.T. Barbaro, Dr. J.S. de Pinho Gonçalves

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Implementation and Evaluation of an Order Parameter for the Reaction-Diffusion Model in a Cellular Automaton

Daniel Puente Barajas 

Delft University of Technology, BSc Computer Science and Engineering, CSE3000 - Research Project
Creation of Gang Territories and other Patterns

Abstract

This paper describes the process and evaluation methods by which we adapted a Reaction-Diffusion model and an order parameter to monitor its segregation state in a 2D Cellular Automaton model. The model simulates Turing pattern formations, whose behavior will be studied by an order parameter formulated for this project. This order parameter acts as an indicator of the mixture state of the world. The implementation process of such a cellular automaton and an order parameter is described in detail, as well as the evaluation of this model and its order parameter. Different evaluation methods will be performed on this order parameter to show it as a valid indicator of the segregation state of the world in this model, as well as showcase different interesting properties and limitations of the parameter. Such methods will involve plots like Evolution, Phase-Transition, or Linear Regression plots, and the paper will conclude by summing up all the work taken into this project to confirm that this order parameter, with its formulation in this paper, is a valid and interesting one to use.

1. Introduction

In 1952, Alan Turing published a paper proposing a model to simulate various patterns in a dynamic world [Tur52]. These patterns, later known as Turing patterns [Ani22], simulated the growth of some agents in a homogeneous world, and mirrored natural phenomena like bacterial growth. This all led, years later, to the emergence of Pattern Formation and Simulation in computer science.

This field investigates the competition among two or more opposing parties for territory, simulating their movements in physical environments and evaluating the outcomes. The applications of this research are extensive, ranging from predicting bacterial growth for improved disease treatment to preventing gang formation and violence in local communities.

To simulate pattern formation, we employ a Cellular Automaton (CA) world where cells represent discrete states, specifically two in our project. These cells, acting as agents, can change over time steps. Further details on Cellular Automata are provided in Section 3. Our paper focuses on studying the complexity generated by these models when simulating Turing patterns, specifically examining the segregation state of the world through an order parameter.

In this context, the idea of an order parameter is introduced. This order parameter, commonly used in physics and science to assess the order and segregation state in phase-transition experiments [CHK*94], finds similar application in our project within the realm of Computer Science. We utilize it to control and analyze the pattern growth simulations in our world, and it serves as an indicator of world segregation, distinguishing whether agents in

the model are surrounded by agents of the same class (segregated state) or agents of a different class (well-mixed state). In Section 2 a detailed explanation can be found.

This research project thus aims to address the question, "How does an order parameter perform on a Reaction-Diffusion model implemented in a Cellular Automata domain?" Consequently, we study the evolution of this order parameter within the Cellular Automata world, verifying its validity and behavior through various evaluation processes. The research question is further divided into sub-questions, which are defined and elaborated in detail in Section 4.

The paper will first discuss previous work in Section 2. After that, Section 3 will show and explain all relevant information to understand the project, which will be followed by Section 4 with a subdivision of the research question, and the methodology by which this question will be answered. Following that, all results obtained to answer the question will be shown and discussed in Section 5. Next, the topic of Responsible Research will be discussed in Section 6, and the paper will end with a conclusion of the research project and its findings in Section 7. The research to conduct this project was conducted in a Jupyter notebook, with some helper Python files, and can all be found in GitHub: <https://github.com/Dpbarajas/Research-Project>

2. Related Work

This section will focus on explaining different papers that implement different order parameters. Traditionally, an order parameter

is represented as a value close to 0 when the world it is monitoring is well-mixed, and close to 1 when it is segregated. This section will focus on researching applications of this parameter outside of the area of Computer Science. The biggest area where this order parameter concept is used is Physics, with different applications in several areas, like Crystallography, with examples like controlling the temperature and the crystallization state of different materials [SSY11] or using order parameters to classify different water structures and ice polymorphs [DTA21].

It is also being used in the Quantum Physics and Mechanics domain since processes in this area make a connection between physical phenomena and quantum computations. Examples of this area include explaining different processes, like simulating quantum-mechanical nematic orders [tVW20] or the mutual information concept through the conventional order parameter in condensed matter physics [GYL13]. These are just some examples where the concept of the order parameter is useful in different experiments outside of Computer Science. With these examples, we can see that it is a very useful parameter in different domains of science and research.

This section will also briefly explain the work that has already been done in the area of Pattern Formation Simulation. It will briefly be discussed since the explanation of the different models and papers that directly contribute to this research project will be given in detail in Section 3, where the Background of the project is explained.

Firstly, an application of a Reaction-Diffusion model to a CA world in three dimensions was found [SP17]. In this paper, an existing Reaction-Diffusion model in 2D [You84] is used and extended to the three dimensions.

Secondly, a convection-diffusion model is introduced, which rules how agents in a lattice move, drop graffiti from their own gang to create their territory and stay away from the graffiti of different gangs. [AB18]. In this paper, a formula for an order parameter as an indicator of the mixed state of the world is postulated. This is where we are first introduced to the concept of the order parameter. In a later paper, this model is extended to an arbitrary amount of gangs and adapts an order parameter accordingly [AB21].

3. Background

This section will explain all the relevant models and concepts that will appear in the project, as well as the formulas that explain them.

The first concept to understand in depth is that of Cellular Automata: they are discrete models that use straightforward rules that execute on a grid of cells to simulate the behavior of complex systems. According to these rules, each cell in the grid can be in one of a finite number of states and is updated in discrete time steps in accordance with these rules. These updates take place simultaneously, which means that each cell's state is updated based on the states of its neighbors in the preceding time step. One of the most famous Cellular Automata is Conway's Game of Life, generalized by Martin Gardner [Gar70] and heavily studied by people all around the world [Joh08]. Cellular automata are valuable for simulating a wide range of phenomena, from physical systems to social

dynamics, because they exhibit complex behavior from a simple set of rules. [Wol02]

The project will use Cellular Automata to implement a Reaction-Diffusion model as a model that generates Turing patterns, implemented by Young [You84] and further defined by Skrodzki in his paper before proposing a modified model to showcase this behavior and the formation of Turing patterns in 3D [SP17]. In a later paper, he studies and classifies the kind of Turing patterns that are formed into the different kinds of shapes generated by the model [SRZ20]. The model used in both papers works by initializing every cell in the grid as a 0 or a 1 with a given probability, conventionally of 0.5, and updating the cells by making use of a circular kernel with an inside and an outside radius r_i and r_o , respectively. We use a kernel for the calculation of the cells' states in the next time step. That kernel is represented as a 2D grid there, given r_i and r_o , the elements inside r_i are set to 1, the cells in the annulus between r_i and r_o are set to -1, and anything outside that is set to 0. At each step t , each cell C in the world uses this kernel to calculate the weight of its neighbors, defined as C_i , as well as its state in the next step $t + 1$ is set according to these parameters and the following equation:

$$s_{t+1}(C) = \begin{cases} 1 & \sum_i C_i \cdot \omega_r(C_i) > 0 \\ 0 & \sum_i C_i \cdot \omega_r(C_i) \leq 0 \end{cases}, \quad (1)$$

$$\omega_r(C_i) = \begin{cases} 0 & (x - x_i)^2 + (y - y_i)^2 > r_o^2 \\ 1 & (x - x_i)^2 + (y - y_i)^2 \leq r_i^2 \\ -1 & \text{otherwise} \end{cases}, \quad (2)$$

However, these papers don't implement any sort of order parameter, which led to the aim of evaluating this order parameter in this setting, and subsequently to the research question postulated in this research project, which will be further developed in section 4. So the next step is to formulate an order parameter and its behavior for this model. An order parameter is represented as a value between 0 and 1, where an output close to 1 means that most of the cells from a class are surrounded by cells of the same class; and an output close to 0 means that most cells of a class are surrounded by cells from a different class. Thus, when this order parameter is 1, the whole world is covered by cells of the same class (segregated), and when this order parameter is 0 the whole world is randomly filled by cells of different classes. This parameter is computed at each time step and it shows the evolution of the world in the behavior of the agents through the simulation.

In the paper "A Multispecies Cross-Diffusion Model for Territorial Development" [AB18], an order parameter formula is proposed given the random-walker model. This model differs from the Reaction-Diffusion model in that it deals with agents in a lattice moving, dropping graffiti from their own gang to create their territory, and staying away from the graffiti of different gangs. In a later paper, the model is extended to an arbitrary amount of gangs and is adapted accordingly [AB21].

An order parameter is also introduced in this paper as an indicator of the mixed state of the world. This is where we are first introduced to this concept in the area of Pattern Formations and Simulations, which uses the information available in the world (the

number of agents of a gang in the lattice, specifically), to showcase the expected segregation. However, the Reaction-Diffusion model does not use the concept of the number of agents in a cell, so the formula posed in the paper is not useful to this project. Thus, a new equation has been formulated making use of the properties and available parameters in the Cellular Automaton domain. Given a world CA , the project's order parameter sums over all points of the world (x, y) and calculates the value of that specific point by taking a look at its neighbors (\tilde{x}, \tilde{y}) , which are specified by $N(x, y)$:

$$\varepsilon(t) = \frac{1}{n^2 \cdot |N(x, y)|} \cdot \left| \sum_{(x,y) \in CA} \sigma(x, y) \sum_{(\tilde{x}, \tilde{y}) \in N(x,y)} \sigma(\tilde{x}, \tilde{y}) \right| \quad (3)$$

where
 n is the length of the grid

$$\sigma(x, y) = \begin{cases} -1 & \text{if } CA(x, y) = 0 \\ 1 & \text{if } CA(x, y) = 1 \end{cases} \quad (4)$$

$$N(x, y) = \begin{cases} N_4(x, y) & = \{(x \pm 1, y), (x, y \pm 1)\} \\ N_{D_{yn}}(x, y) & = \{(\tilde{x}, \tilde{y}) \mid (x - \tilde{x})^2 + (y - \tilde{y})^2 \leq r_i^2\} \end{cases} \quad (5)$$

We utilize two neighborhood types, denoted as $N_4(x, y)$ and $N_{D_{yn}}(x, y)$. The former considers the four closest neighbors (top, bottom, left, and right) in its formula, while the latter takes into account neighbors within a circle of radius r_i . By plotting both neighborhoods, we can effectively compare them and determine which one is more suitable for our research computations.

We evaluate and compare these approaches to identify their respective performance. They are referred to as the "Four-neighbor" and "Dynamic-neighborhood" settings, respectively.

Furthermore, Professor Skrodzki utilizes an existing implementation of a general Cellular Automata described in Chapter 6 of the book "Think Complexity" [Dow12]. This implementation includes a Python notebook and accompanying files that showcase simulation behavior and visualization methods. Our research project builds upon this implementation, adapting it to meet the specific requirements of our model.

4. Methodology

This section will discuss the steps taken into answering the research project and its research question. The project aims to answer the question "How does an order parameter perform on a Reaction-Diffusion model implemented in a Cellular Automata domain?". To answer the question, several subquestions are formulated, which will divide the project into its different sub-questions, each taking a subsection:

4.1. Model Creation

The first step in the assessment of the question is to create a model that can simulate Turing pattern formations according to the formulas given by Equations 1 and 2, as well as calculate $\varepsilon(t)$ according to the Equations 3 and 4. This model will be used in a periodic world, meaning that the cells on one side of the world will be counted as

neighbors of the cells on the other side, making the world a never-ending world where the top and left parts of the world can respectively affect the bottom and right parts, and vice versa.

The implementation for this model is developed in Python, in a Jupyter notebook which allows running the code swiftly in separate blocks to separate the computation of different parts of the project. The base modules and methods provided by Think Complexity [Dow12] are given in Python files.

The project implements a `CA` class that performed the update rules according to the formulas defined. When a simulation is to be run, an object is created with the following parameters and properties:

- `n` is the length of the world. The Cellular Automata will be represented as a 2D array of size $n \cdot n$.
- `ri` and `ro` are the kernel radii values, which indicate the number of neighbors that there will be when computing the next state.
- `random` is a Boolean value that expresses how the world will be initialized. If true, each cell in the world will be initialized to 0 or 1 with the attribute `probabilities` defining how likely the cell is to be each value. Else, an arbitrary patch of 20*20 cells will be set to 1 in a random position in the world, while the rest of the world is set to 0. The default value is set to True.
- `static_neighbors` is another Boolean value. If true (the default value), we will use four neighbors to calculate $\varepsilon(t)$. Otherwise, we will use as many cells as there are inside the circle of radius r_i .
- `probabilities` only comes into play when `random` is set to True, and it is a floating point number that expresses the probability that the cells in the world are randomly initialized to 0. The probability that the cells are initialized to 1 is set as $1 - \text{probabilities}$. Its default value is 0.5.

Inside the `ReactionDiffusion` class also lies the implementation for the calculation of $\varepsilon(t)$. At each step, both the next state of each cell as well as the value of $\varepsilon(t)$ are computed in the `step()` method:

```

1 def step(self):
2     # Calculation of the next state of each cell
3     # in the world according to equation (1)
4     update_rule = correlate2d(self.array, self.
5     kernel, mode='same', boundary='wrap')
6     self.array = (update_rule > 0).astype(np.
7     uint8)
8
9     # Implementation of the formula specified for
10    # order parameter in equation (3).
11    self.order_array = (self.array - 0.5) * 2
12    self.order_parameter = np.abs(np.sum(self.
13    order_array * correlate2d(self.order_array,
14    self.order_kernel, **self.options)) / (self.
15    neighbours * (self.n**2)))
16    self.order_list.append(self.order_parameter)

```

Listing 1: Update formula for the CA

We use two different types of kernels in the computation: the update rule kernel, which calculates the next step of the world; and the order parameter kernel, which is used for the $\varepsilon(t)$ formula calculation. This second kernel will be a different one according

to *static_neighbors*, as explained in the parameter list above. Both kernels are computed and generated in the method `kernel_creator()`.

These steps make up the first part of the research, which provides a working model that shows Reaction-Diffusion behavior which will be shown in section 5. In Section 5.1, there are some examples where it is shown that this model does indeed work for some values of r_i and r_o , but not for others.

The next step was to find out for which values the model shows interesting Turing patterns, and how we can do that with $\epsilon(t)$, which was complemented by the evaluation of this order parameter after this, to showcase the correctness of the formula.

4.2. Evaluation of Order Parameter

Following the creation of the model, the project led to evaluating whether this model and $\epsilon(t)$ were correctly implemented. This would tell us if the model was as well correct and if the Equation 3 proposed for it worked. This evaluation is divided into several subsections that implement different tests to evaluate all aspects of the project:

4.2.1. Phase-Transition plots

As shown in the results obtained in Section 5.1, not all values of r_i and r_o output interesting Turing patterns, where the state of the world is not completely segregated after reaching convergence. The hypothesis as to why this happened was that, for the patterns to happen, the area covered by the inner circle and the outer annulus of the kernel used in the update formula had to be relatively similar. So, given two radii r_i and r_o :

$$\begin{aligned}\pi \cdot r_i^2 &\approx \pi \cdot r_o^2 - \pi \cdot r_i^2 \\ \Leftrightarrow 2 \cdot r_i^2 &\approx r_o^2 \\ \Leftrightarrow r_o &\approx \sqrt{2} \cdot r_i\end{aligned}$$

To test this hypothesis, we evaluate all possible values for r_i and r_o from 1 to 20 in a 100x100 simulation. We chart the values that result in a non-segregated state to examine if they follow the equivalence relation. This helps us identify uninteresting patterns by utilizing a Phase-Transition plot. This plot is a 3D representation where the x and y axes represent the values of r_i and r_o respectively, while the z-axis represents the final value of $\epsilon(t)$ in the simulation for that specific position in the lattice. To enhance visualization, cases where $\epsilon(t)$ is 1 are set to 1 since predominantly segregated states would obscure other cases. It's worth noting that $\epsilon(t)$ will never be 0 at the end of any simulation since the world always converges to a different order parameter as it never remains well-mixed.

Additionally, the number of different settings (besides n , r_i and r_o) are whether the world starts randomly or not, and whether $\epsilon(t)$ neighbors are set to 4 or to the cells inside the circle with radii r_i . Thus, in this step and in the next ones until expressed differently, the project will chart all possible combinations of these settings, in this case, four. There will be thus four plots, showcased in Figure 5 in Section 5.2.1.

4.2.2. Linear Regression Plots

We appreciate in the previous Figure 5 that, for the four pairs of settings, most of the pairs of r_i and r_o that provide a non-segregated state follow a sort of line shape. This line, as shown in the previous section, is theorized to follow the line $y = \sqrt{2} \cdot x$, where x is the r_i and y is the r_o . In this section, we plot all the points that have been achieved by the previous evaluation and chart them in a 2D plot against the line $y = \sqrt{2} \cdot x$. We also take notes of the furthest point from the line to get an indicator of how different the point can be from the line, given that it is a discrete domain and the line doesn't stop in discrete points. A plot is generated for each of the four cases, and they are shown in Figure 6, as well as explained in the corresponding subsection in Section 5.2.2.

4.2.3. Evolution Plots

After performing these tests for $\epsilon(t)$, we see that it provides the expected behavior and it is very useful in explaining the segregation state of the world and the presence or not of different patterns. Another interesting concept to learn from this $\epsilon(t)$ is to know how it evolves throughout the simulation, and that is why the next test performed on this simulation is the evolution plots. Those plots chart, for the four different parameter combinations, a random subset of different radii sizes and assess the evolution of $\epsilon(t)$ over the different initialization.

In order to palliate the fact that these simulations involve some sort of random initialization, each simulation is performed 10 times and aggregated throughout the different iterations. Two different methods of aggregation are shown, the average and median, to see if there is any difference between them and assert which one is better. The evolution plots and their discussion are shown in Section 5.2.3, in Figure 7

4.3. Asserting Validity of Model for Bigger r_i and r_o

It has been shown through Figure 6 that, except for some edge cases, the assumption that points following line $y = \sqrt{2} \cdot x$ provide interesting patterns -Turing patterns when the world is initialized randomly-. It was important, however, to show that this assumption held for bigger values of r_i and r_o . In this section, that question is tackled by performing a simulation with bigger parameters according to this line, and checking that Turing patterns are still formed in Figure 8, as well as that $\epsilon(t)$ remains within the behavior established. It is all showcased in Section 5.3.

4.4. Limitations of Order Parameter

The previous sections, as well as their corresponding results, tell us that $\epsilon(t)$ is a good order parameter that keeps track of the segregation state of the world. This can be very helpful for some aspects of Pattern Simulation, like calculating when the world has reached a convergence state or checking which values of r_i and r_o provide Turing and other kinds of patterns. This subsection, however, focuses on a new aspect of these patterns for which the concept of an order parameter doesn't provide useful information.

This specific aspect will consist of a Turing Pattern Classification experiment where different kinds of Turing patterns are attempted to be classified according to the shape that they generate.

Something similar was performed in 3D, in his paper "Visually-guided investigations of sub-structures in 3d Turing - like patterns." [SRZZ20]. This research paper will perform a similar experiment.

First of all, the different Turing pattern shapes that have been appreciated in the simulations are presented in Figure 1. These different shapes each happen with different pairs of r_i and r_o with random initialization, and its order parameter is assessed with the N_4 neighborhood.

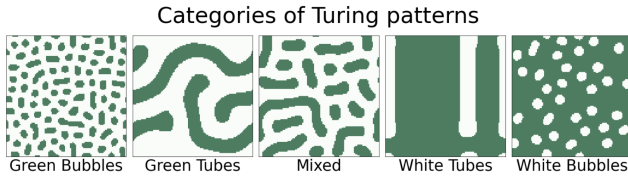


Figure 1: Different classifications of Turing patterns in 2D; each of them is an example of the kind of patterns that belong in that subgroup along with the name given to them.

Once the different categories are specified, the experiment was formulated as taking the different valid r_i s and r_o s obtained in the first plot of Figure 5, since this experiment was performed with the traditional parameters -random initialization and N_4 -, and after simulating the convergence states for each of these states, classifying each of these final states into each of the corresponding categories. For the purpose of stabilization against randomness and increased amount of points, each pair of valid parameters is used in the simulations a total of 5 times.

After classifying the different simulations, a box and violin plot is generated for each of the categories. First, in Figure 9, $\epsilon(t)$ is used in the plots, and it is explained why this doesn't work in Section 5.4. After that, in Figure 10 the plot is generated with the concept of Relative Area, which is given as the number of activated cells divided by the total number of cells in the world. We further explain the results in Section 5.4.

5. Results

This section will show the results obtained for each of the sub-questions for the postulated research question, where these results and plots will be explained and discussed. It follows the same structure as Section 4.

5.1. Model creation

After implementing the model, it was appreciated that, for different r_i and r_o , not all values generated Turing patterns, as seen in Figure 2.

However, for certain arbitrary values, intriguing patterns emerged, revealing an intermediate convergence state where $\epsilon(t)$ ranged between 0 and 1. This is illustrated in Figures 3 and 4. The variation in patterns between the figures stems from different initializations. Random initialization yields diverse Turing patterns as the radii sizes change, while segregated initialization produces distinct patterns with unique behaviors, as depicted in Figure 4.

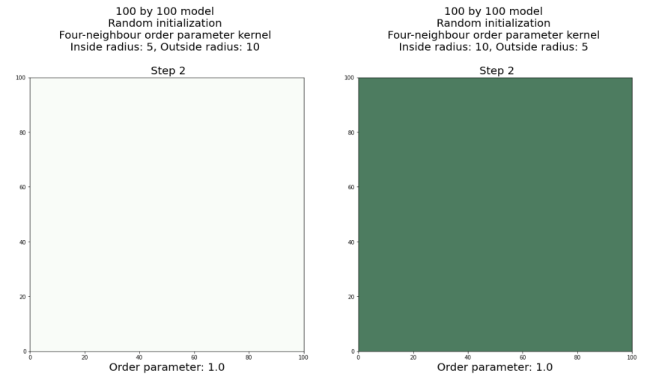


Figure 2: Example simulation where no patterns are formed and fully converged states are reached at the first step. Both worlds are set with size 100x100, with random initialization and a four-neighbor order kernel. On the left image, r_i is set to 5, and r_o is set to 10, and on the right, r_i is set to 10, and r_o is set to 5, and on the right. $\epsilon(t)$ for both cases is 1 since they are both completely segregated states.

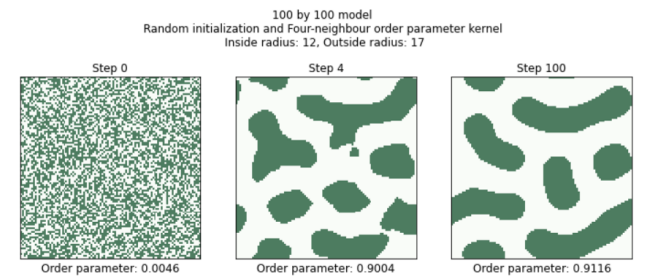


Figure 3: Example simulation that shows Turing pattern formations with the Reaction-Diffusion model. The world is set to size 100x100, with random initialization and a four-neighbor order kernel. The computation kernel is set to $r_i = 5$ and $r_o = 10$.

These figures demonstrate that the model produces both Turing and non-Turing patterns, depending on the initialization state, for specific combinations of r_i and r_o in each case. Thus, not all input pairs yield interesting patterns; only certain values do. Determining these values became the focus of the subsequent phase in the research project: evaluating $\epsilon(t)$. Section 4.2 elaborates on the rationale behind this investigation.

5.2. Evaluation of order parameter

The evaluation of $\epsilon(t)$ and its functionality is divided into three subsections: Phase-Transition plots, Linear Regression plots, and Evolution plots.

5.2.1. Phase-Transition plots

The four plots shown in Figure 5 show a clear trend of Turing patterns forming in a sort of line trend, which will be explored more in-depth in the following subsection.

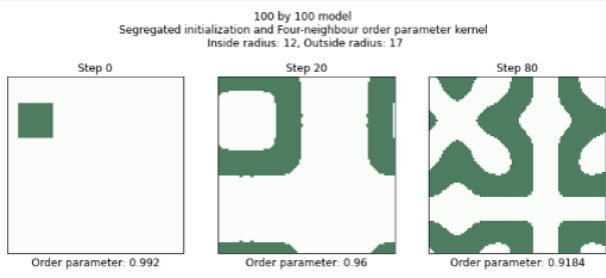


Figure 4: Example simulation that shows pattern formations with the Reaction-Diffusion model. The world is set to size 100×100 , with segregated initialization of a 20×20 patch of activator in a random place and a four-neighbor order kernel. The computation kernel is set to $r_i = 12$ and $r_o = 17$.

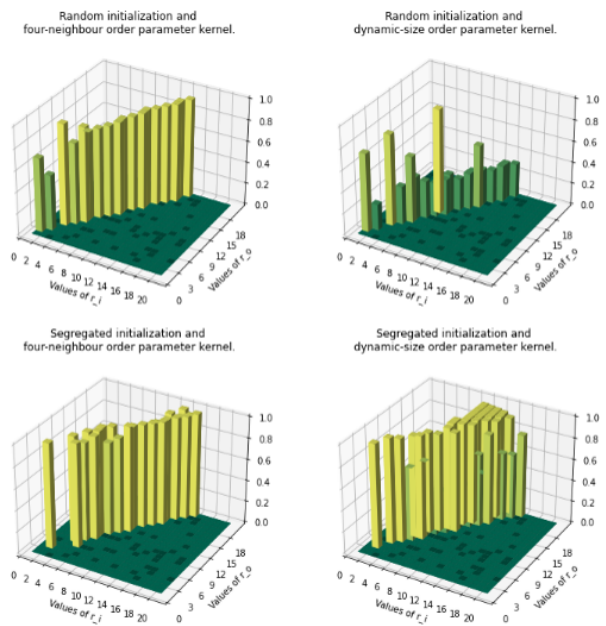


Figure 5: Example simulation that shows Turing pattern formations with the Reaction-Diffusion model. The world is set to size 100×100 , with random initialization and a four-neighbor order kernel. The computation kernel is set to $r_i = 5$ and $r_o = 10$.

However, this line trend is broken in the segregated case with $N_{D_{\text{dyn}}}$, where a sort of triangular shape appears as r_i and r_o grow in size. This can be explained by the fact that, as these parameters increase, more and more cells are used for the computation of the next steps in the update formula for the model 11, which leads to the behavior of the world becoming broader and more chaotic. As a consequence of this, the final shapes are not completely segregated but don't belong to the category of interesting patterns, they are just blobs of deactivated or activated cells that don't provide much insight into the topic at hand. However, $\epsilon(t)$ is only set to 0 if the world is completely segregated, so these kinds of patterns were

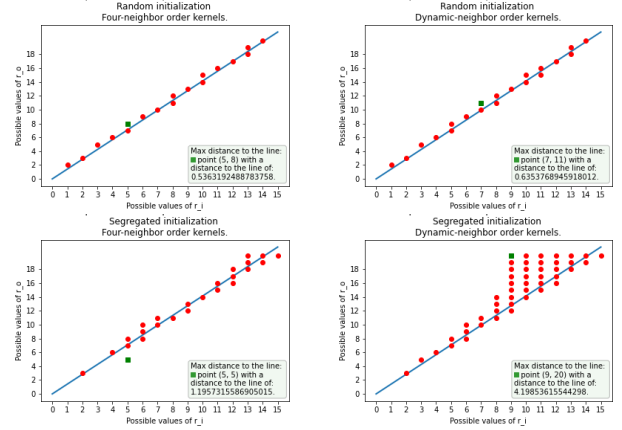


Figure 6: Linear regression plot of the valid parameters in each initialization setting. The line the points are plotted against is given as $y = \sqrt{2} \cdot x$, and the point that is the furthest away from this line is also shown for each of the plots.

present in the graph. For the experiments where the shapes matter, these pairs of radii that don't provide useful patterns are ignored.

We can appreciate that, in both cases where the dynamic neighborhood is used, and more specifically in the case with the random initialization, $\epsilon(t)$ ranks way lower and more chaotically than in the case with four neighbors. This leads us to confirm that, regardless of the initialization setting, the dynamic neighborhood acts as a worse calculator for $\epsilon(t)$. This can be explained by taking into account that the number of neighbors used grows as r_i and r_o grow, which takes into account increasingly more cells and creates a worse order parameter: even if it still ranges from 0 to 1, the same shapes in random and segregated initializations will have a lower order parameter, so there will be a big jump in this parameter between almost segregated states and completely segregated ones.

Thus, we conclude that using four neighbors to calculate $\epsilon(t)$ is a better practice for random and segregated initializations since it provides better and higher performing values.

5.2.2. Linear regression plots

As the plots in Figure 6 clearly show, the pairs of points that provide non-segregated states clearly follow a trend that is explained by the blue line $r_o = \sqrt{2} \cdot r_i$. However, the line doesn't provide discrete values, so the red points are scattered around this line, in discrete coordinates in the graph. That is why it was interesting to know what the furthest point from this line could be, as well as its corresponding value for $\epsilon(t)$. That is what is described in the green text boxes. This is an indicator of how far away the values pairs of radii can be from the line while still providing non-segregated patterns, which can be useful when scaling the model to higher values. An example of this is explained in Section 4.3 and shown in Section 5.3.

As previously mentioned, the lower right plot is filled with false positives, which reach a convergence state that just consists of cases like ones with a bubble in the middle of a segregated state. Future

work could be destined to a sort of false positive detection system in this setting since they had to be removed manually for the following sections.

5.2.3. Evolution plots

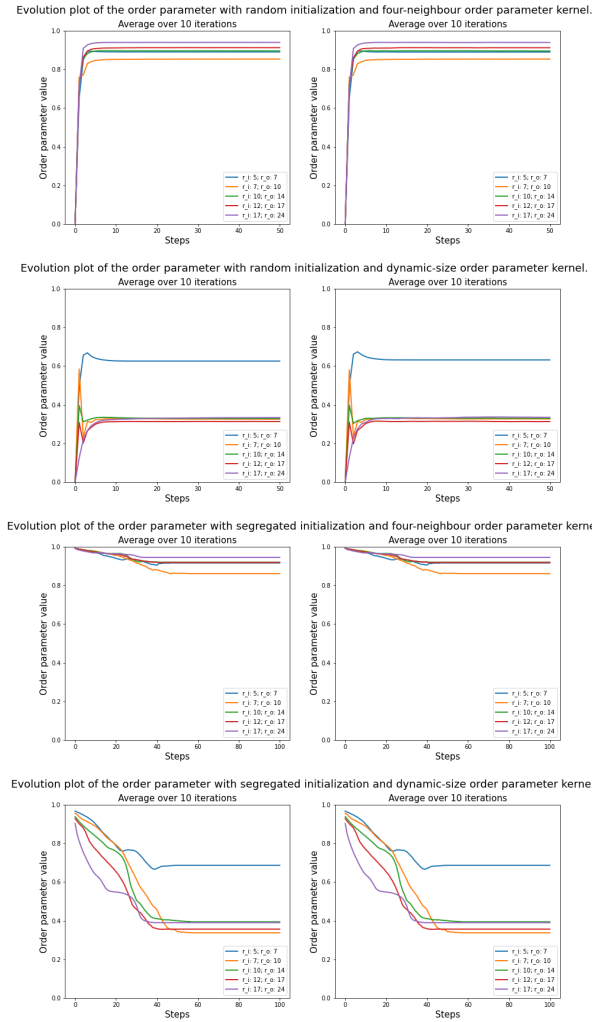


Figure 7: 10-iteration Evolution plots for the four different initialization settings posed in this project. Each of the settings has two graphs, the left one with average aggregation and the right one with median aggregation. A random subset of 5 valid pairs of radii is used.

The first thing that can be appreciated in the plots from Figure 7 is that the median and average plots look exactly alike. This leads to believe that $\epsilon(t)$ won't change regardless of the aggregation method implemented, so the average aggregation method is chosen as most experiments use this method for any sort of similar calculations. The median method was proposed as an alternative at some point in the project, but the fact that there is no difference between them makes it non-eligible as the preferred aggregation method.

This plot shows a general behavior in line with what is expected

from $\epsilon(t)$: when the world is started with the random initialization, $\epsilon(t)$ starts at 0 and rises quickly to then converge on a number between 0 and 1. For the segregated initialization setting, the opposite happens, since the starting state, with a mostly segregated world, starts at 1 and, more slowly than in the random case, it approaches a number between 0 and 1 as well.

There is also a clear distinction between the 4-neighbors (N_4) setting and the dynamic neighbors ($N_{D_{dyn}}$) setting. This distinction is shown as a way more random and chaotic behavior in the $N_{D_{dyn}}$ setting, which is explained by the fact that, as more neighbors are considered for the $\epsilon(t)$ computation, the behavior of this parameter is dictated by more variables, which ranks it lower than (N_4) in cases like small segregated states, where (N_4) and ($N_{D_{dyn}}$) rank it as a positive and negative contribution to the formula, respectively.

In fact, the behavior exhibited in the first plot is the same one as the one exhibited in Figure 5 of the paper where an order parameter was first found [AB18], where a different order parameter with different parameters is studied. This leads to the belief that the formula for $\epsilon(t)$ implemented in this project (3, 4) is correctly translated from that paper since, for the same settings, the behavior is the same.

5.3. Asserting validity of model for bigger r_i and r_o

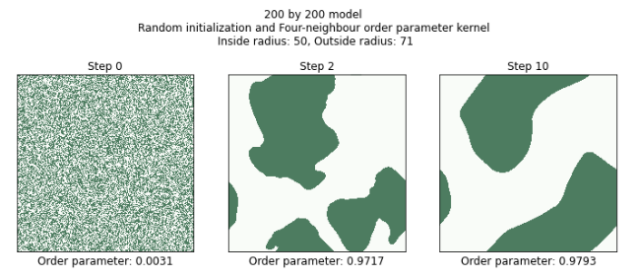


Figure 8: Simulation with bigger parameters: $n = 200, r_i = 50$ and $r_o = \sqrt{2} \cdot 50 \approx 71$. Random initialization with the use of four neighbors for the $\epsilon(t)$ computation.

Figure 8 is an example of the model working and providing Turing patterns for bigger r_i and r_o , with these values being related to each other by the equality $r_o = \sqrt{2} \cdot r_i$. This proves that the hypothesis postulated in Section 4.2.1 is correct in a more general case than the original simulations, which range the radii values with $n = 100$ and r_i having a maximum value of 20.

In this case, r_i is set to 50 and thus r_o is set to 71, the closest integer to $\sqrt{2} \cdot 50 = 70,7106\dots$. We need integers in our calculations of the radii, which is why the value is rounded up to 71. However, this means that we get a kernel that is bigger than the world, which means that, since the world is periodic, the radii will cover the same cells twice, which will lead to the model breaking. That is why we need to increase our n accordingly.

The maximum values of the radii for a world of $n = 100$ are 20, so the ratio of r_i to n is a maximum of $1/4$. We will use a size of n that keeps this ratio going: so, given an arbitrarily chosen r_i , we will choose $n = 4 \cdot r_i$. Since our chosen r_i is 50, then $n = 200$. These assumptions make the model hold.

5.4. Limitations of order parameter

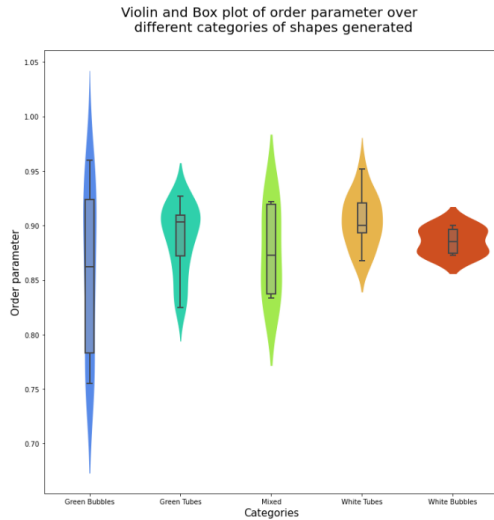


Figure 9: Violin and Box plot of the different shapes against $\epsilon(t)$ in the y-axis. A total of 80 simulations are used, with 5 iterations of a subset of 16 valid parameters.

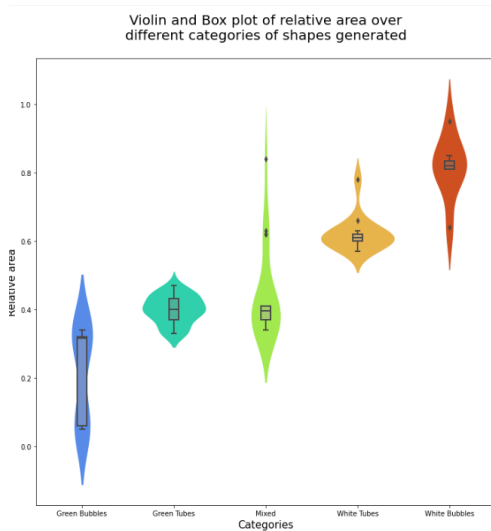


Figure 10: Violin and Box plot of the different shapes against the relative area of that simulation in the y-axis. A total of 80 simulations are used, with 5 iterations of a subset of 16 valid parameters.

Figure 9 reveals that $\epsilon(t)$ is insufficient for classifying different Turing patterns into distinct shapes. $\epsilon(t)$ fails to differentiate between mostly activated and mostly deactivated worlds, only their segregation. Additionally, all order parameters are between 0.7 and 1. To address these limitations, a new parameter, the Relative Area, is proposed. It is calculated by dividing the number of activated cells by the total number of cells in the world. Figure 10 demonstrates that the Relative Area serves as a superior indicator for the diverse shapes generated in the simulations. This improvement

stems from its ability to discern mostly deactivated worlds (low area) from mostly activated worlds (high area), a crucial feature for this experiment. Hence, the Relative Area can be utilized to identify the specific type of Turing shape observed in the simulations, which can be used in later experiments as a predictor without the need for manual classification.

6. Responsible Research

This section will discuss the morality of this research project, and it will be divided into two sections: the ethical aspects of the project and the reproducibility of the experiment.

6.1. Ethical aspects

Ethics are an important part of computer science and its development. Computer science has a profound impact on society nowadays and it is very important to incorporate ethical considerations into the development and use of technology, to ensure the correct evolution of this area in different aspects of human morals and values.

This research project focuses on the simulation of different patterns and their control through the use of a parameter. On the surface, the process seems harmless enough, but when looking for applications for these concepts, they can become harmful in a distinct variety of ways.

Since this model can reproduce patterns that happen in bacteria growth and systems in the natural world, the understanding of these processes can lead to the creation of biological hazards in the form of new viruses and diseases that behave differently from the previous ones. Bacteria are already showing growing resistance to antibiotics, as the more these medicines are used and misused, these microorganisms learn to resist the medicines, and it is a problem that will affect a lot of people in the future since diseases that are cured easily nowadays will become a difficult process to heal from [Org21]. The understanding of the way they grow and reproduce can be used for good, by helping eradicate these diseases, or, as mentioned before, for bad to potentiate these kinds of organisms.

Another ethical aspect to take into account is the gang simulation part of this project. Some models used in similar projects are used to simulate how agents in different gangs would move and create their territories. This can be used by the police and other forces to predict and prevent gang violence. However, models like this have already been tried and they have shown a biased prediction, which in turn has brought about more violence and rejection against the police in the groups that have been discriminated against by this model. An example of this has happened in Los Angeles, where the LAPD had to turn off the implemented model, which used data from previous criminal activities and AI to predict crime in the city, due to these kinds of problems [Bhu21]. If it is decided to keep investigating this application of gang simulation, a thorough investigation should be performed before applying it to the real world, since we need to ensure the fairness and correctness of these models for the well-being of all society.

In summary, gang and bacteria simulation has very interesting

and debatable ethical ramifications that should be studied and assessed in depth, since it is only in that way that computer scientists can ensure responsible practices that look for the collective good of the people. By integrating ethics into computer science, professionals can make informed decisions, comply with legal requirements, and contribute to a sustainable and beneficial technological landscape for everyone.

6.2. Reproducibility

Reproducibility refers to the concept of being able to repeat the steps taken in any research, and it is a very important aspect of experimental processes since it diminishes the possibility of fake data being introduced and used in order to fake results. During one of the Responsible Research lectures provided by TU Delft in the scope of this project, we were introduced to a case where Nobel-winning scientist Robert Millikan cherry-picked data points from a pool of data and dropped others in order to fit his hypothesis in a study about electron charges [PG00]. This was discovered by Felix Ehrenhaft after he tried to reproduce his work, and found out that Millikan had faked his data and his work was thus incorrect and fraudulent.

In the case of this project, no experiments or surveys have been needed to perform the research, so the reproducibility is exclusively related to the availability of the code used for our experiments. That is why all files and code used will be made available for everyone to have access to, be able to run the same simulations as the ones taken in this project, and build upon the codebase to try different methods and evaluation processes. This should cover the reproducibility aspect of the project in a concise and precise enough way, to show that no data or fact in this project has been falsified or modified.

7. Conclusion

The research project aimed to implement a functional order parameter for assessing the state of the world in the Reaction-Diffusion model for Cellular Automata. The main research question was: "How does an order parameter perform on a Reaction-Diffusion model implemented in a Cellular Automata domain?" To address this question, the project was divided into four sub-questions: model creation, model and order parameter evaluation, assessing the model's scalability, and exploring the limitations of $\epsilon(t)$. These sub-questions collectively formed a comprehensive research project.

The paper used a systematic structure to answer the question. It began with an introduction (Section 1) that outlined the concepts and the research question, as well as the structure of the paper. It then delved into related work (Section 2) to explore existing order parameters. The background (Section 3) introduced the model and formulas used in the research. The methodology (Section 4) detailed the approach taken to answer the research question, including the subdivision into sub-questions, the evaluation methods employed, and the rationale behind their selection. The results (Section 5) presented plots and graphs that addressed the sub-questions outlined in the methodology, accompanied by in-depth explanations. The responsible research section (Section 6) discussed the

ethical and reproducibility aspects of the experiment, ending in this conclusion (Section 7).

Based on the evaluation methods and plots, $\epsilon(t)$ was deemed a reliable indicator of the world's segregation state. Additionally, using N_4 instead of $N_{D_{\text{dyn}}}$ as neighborhoods reduced irregular behavior (Figure 5). Later it was seen that $\epsilon(t)$ successfully identified valid parameters for generating interesting non-segregated patterns along a specific line (Figure 6) and his evolution over time was shown (Figure 7). Furthermore, it was observed that the hypothesized line's points generated Turing patterns as the domain size increased (Figure 8). The limitations of $\epsilon(t)$ were explored through a Turing Shape Classification experiment, which prompted the introduction of Relative Area for shape classification (Figures 9 and 10).

Following this research, this area of computer science offers a wide range of opportunities, such as extending the model and $\epsilon(t)$ formula to accommodate multiple classes, investigating the impact of different initialization settings, or exploring the effect of changing initial probabilities. These examples highlight the potential for enhancing this research with additional work in the future.

References

- [AB18] ALSENAFI A., BARBARO A. B.: A convection–diffusion model for gang territoriality. *Physica A: Statistical Mechanics and its Applications* 510 (2018), 765–786. URL: <https://www.sciencedirect.com/science/article/pii/S0378437118308604>, doi:<https://doi.org/10.1016/j.physa.2018.07.004>. 3, 8
- [AB21] ALSENAFI A., BARBARO A. B. T.: A multispecies cross-diffusion model for territorial development. *Mathematics* 9, 12 (2021). URL: <https://www.mdpi.com/2227-7390/9/12/1428>, doi:[10.3390/math9121428](https://doi.org/10.3390/math9121428). 3
- [Ani22] ANIRBAN A.: 70 years of turing patterns. *Nature Reviews Physics* 4 (06 2022), 1–1. doi:[10.1038/s42254-022-00486-8](https://doi.org/10.1038/s42254-022-00486-8). 2
- [Bhu21] BHUIYAN J.: Lapd ended predictive policing programs amid public outcry. a new effort shares many of their flaws, Nov 2021. URL: <https://www.theguardian.com/us-news/2021/nov/07/lapd-predictive-policing-surveillance-reform>. 9
- [CHK*94] CLARKE J. B., HASTIE J. W., KIHNBORG L. H. E., METSELAAAR R., THACKERAY M. M.: Definitions of terms relating to phase transitions of the solid state (iupac recommendations 1994). *Pure and Applied Chemistry* 66, 3 (1994), 577–594. doi:[doi:10.1351/pac199466030577](https://doi.org/10.1351/pac199466030577). 2
- [Dow12] DOWNEY A.: *Game of Life*. O'Reilly Media, 2012, ch. 6, p. 89–102. URL: <https://books.google.nl/books?id=SaV59o7K0GMC>. 4
- [DTA21] DOI H., TAKAHASHI K. Z., AOYAGI T.: Searching for local order parameters to classify water structures at triple points. *JOURNAL OF COMPUTATIONAL CHEMISTRY* 42, 24 (SEP 15 2021), 1720–1727. doi:[10.1002/jcc.26707](https://doi.org/10.1002/jcc.26707). 3
- [Gar70] GARDENER M.: The fantastic combinations of john conway's new solitaire game "life". In *Mathematical games* (1970). 3
- [GYL13] GU S.-J., YU W. C., LIN H.-Q.: Construct order parameters from the reduced density matrix spectra. *ANNALS OF PHYSICS* 336 (SEP 2013), 118–129. doi:[10.1016/j.aop.2013.05.014](https://doi.org/10.1016/j.aop.2013.05.014). 3
- [Joh08] JOHNSTON N.: Conwaylife.com, Dec 2008. URL: <https://conwaylife.com/>. 3

- [Org21] ORGANIZATION W. H.: Antibiotic resistance, Jul 2021. URL: <https://www.who.int/news-room/fact-sheets/detail/antibiotic-resistance>. 9
- [PG00] PRITCHARD M., GOLDFARB T.: The Millikan Case - Discrimination Versus Manipulation of Data | Online Ethics, 2000. URL: <https://onlineethics.org/cases/ethics-science-classroom/millikan-case-discrimination-versus-manipulation-data>. 10
- [SP17] SKRODZKI M., POLTHIER K.: Turing-like patterns revisited: A peek into the third dimension. In *Bridges 2017: Mathematics, Art, Music, Architecture, Education, Culture* (01 2017). URL: https://www.researchgate.net/publication/319879343_Turing-Like_Patterns_Revisited_A_Peek_Into_The_Third_Dimension. 3
- [SRZ20] SKRODZKI M., REITEBUCH U., ZIMMERMANN E.: Experimental visually-guided investigation of sub-structures in three-dimensional turing-like patterns, 09 2020. doi:10.13140/RG.2.2.27508.07043. 3,6
- [SSY11] SIMOES M., SIMEAO D. S., YAMAGUTI K. E.: Global behaviour of the order parameter in nematic phase. *LIQUID CRYSTALS* 38, 8 (2011), 935–941. doi:10.1080/02678292.2011.587131. 3
- [Tur52] TURING A. M.: The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences* 237, 641 (1952), 37–72. URL: <http://www.jstor.org/stable/92463>. 2
- [tVW20] TE VRUGT M., WITTKOWSKI R.: Orientational order parameters for arbitrary quantum systems. *ANNALEN DER PHYSIK* 532, 12 (DEC 2020). doi:10.1002/andp.202000266. 3
- [Wol02] WOLFRAM S.: *A New Kind of Science*. Wolfram Media, May 2002. URL: <https://www.wolframscience.com>. 3
- [You84] YOUNG D. A.: A local activator-inhibitor model of vertebrate skin patterns. *Mathematical Biosciences* 72, 1 (1984), 51–58. doi: [https://doi.org/10.1016/0025-5564\(84\)90060-9](https://doi.org/10.1016/0025-5564(84)90060-9). 3