# PROJECT

Fabian Dacic, Yuyao Duan

May 3, 2021

## 1 LINEAR RECURRENCE RELATIONS

According to the task, we can identify this math problem as belonged to linear recurrence relation. The checkerboard is consisted of $2 \times n$ where $n \geq 1$, therefore $a_1 = 2 \times 1$ checkerboard, $a_2 = 2 \times 2$ checkerboard, $a_3 = 2 \times 3$ checkerboard, $a_4 = 2 \times 4$ checkerboard ... $a_n = 2 \times n$ checkerboard.

For $a_1$ there is only 1 possibility to cover this checkerboard by using two $1 \times 1$ squares, for this reason $a_1 = 1$ i.e. there is only one way to cover it. Based on this understanding, $a_2$ can be covered by two different strategies: 1 L-shaped figure and a $1 \times 1$ square or four $1 \times 1$ squares. By implementing the first strategy, there are 4 ways to cover this $2 \times 2$ checkerboard by rotating the L-shape in 4 different ways; the second strategy, using four $1 \times 1$ square, can be counted as another way to cover since they are all the same. In total, there are 5 ways to cover $a_2$, therefore $a_2 = 5$. Continuing with the logic of $a_2$, $a_3$ can be covered by three different strategies: 1 L-shaped figure, 2 L-shaped figures, or six $1 \times 1$ squares. By implementing 1 L-shaped figure strategy, there are 8 ways of covering this checkerboard due to rotation of the figure in the checkerboard; 2 L-shaped figures strategy has 2 different ways of covering checkerboard since the 2 L-shaped figures can change positions; using six $1 \times 1$ square is 1 way of covering the checkerboard, therefore in total $a_3 = 11$. Persisting with the understanding and strategy of $a_3$, 33 different ways to cover $a_4$ and 87 different ways to cover $a_5$ have been identified.

The conclusion for the initial stage: $a_1 = 1$, $a_2 = 5$, $a_3 = 11$, $a_4 = 33$, $a_5 = 87$. To inference the recurrence relations for $a_n$, we can see that to exactly fill the $2 \times n$ checker-board so that the first rows consisting of n-1 are rectangles (meaning that a $2 \times 1$ checker-board gives us two 1x1 squares to fill it), four different ways to fill the first rows consisting of n-2 that contain 1 L-shaped figure and aforementioned rectangles and two ways to fill the first n-3 rows with w L-shaped figures, therefore we propose the relation as $a_n = a_{n-1} + 4a_{n-2} + 2a_{n-3}$.

Testing $a_4 = 33$, we found that $33 = 11 + 4 \times 5 + 2 \times 1$ which fits the formula; $a_5 = 87$ which is actually $33 + 4 \times 11 + 2 \times 5$ also fulfilling the deduction.

Based on these understanding, we applied Mathematica to explore this task. The solution is as following: first of all, three initial variables are set up which are 1, 5, 11; after this, the formula $a_n = a_{n-1} + 4a_{n-2} + 2a_{n-3}$ is used for this recurrence relation; then applying RSolve to identify the recurrence relations and find the general term formula. By printing out some values of $a_n$ could confirm the the previous inference.

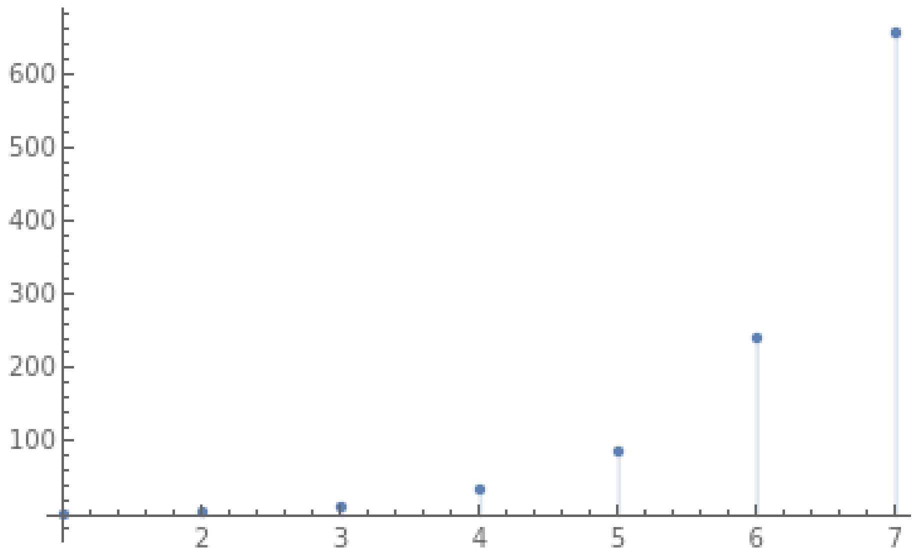


Figure 1.1: The first seven values of the recurrence relations

As we can see from Figure 1.1, Mathematica shows that $a_1 = 1$, $a_2 = 5$, $a_3 = 11$, $a_4 = 33$, $a_5 = 87$, $a_6 = 241$, $a_7 = 655$ which is coherent with above findings. The general term formula of $a_n$ is as following:

$$a_n = \frac{3(-1)^n(5+3\sqrt{3}) - (1-\sqrt{3})^n(9+5\sqrt{3}) + (1+\sqrt{3})^n(9+5\sqrt{3})}{15+\sqrt{3}} \tag{1.1}$$

# 2 Nonlinear Recurrence Relations

According to the task, we can identify this discrete math problem which is belonged to nonlinear recurrence relation–logistic map. The equation to explore in this task is $g(x) = asin(\pi x)$ which the starting value is between 0 and 1 and a is also between 0 and 1.

The principle of this equation is same as the original $x_{n+1} = ax_n(1 - x_n)$ which a (or r) is normally between 0 and 4. Along with changing value of a, the logistic map will show different period doublings until chaos, typically when a is between 3 and 4.

By implementing Mathematica, the logistic map of this new equation can be plotted and Feigenbaums constant can be estimated according to the coordinates of the diagram. To retrieve the new logistic map, the following codes are made changes to adapt for this task:

```
g[x_]:= a (Sin[Pi*x]);
...
a = 0.7; start = 0.1; iter = 50;
...
```

It is worth to mention that, in order to clearly investigate the patterns of doubling periods, there are several diagrams plotted by using different *a* values.

The period doublings can be observed by the following diagram:

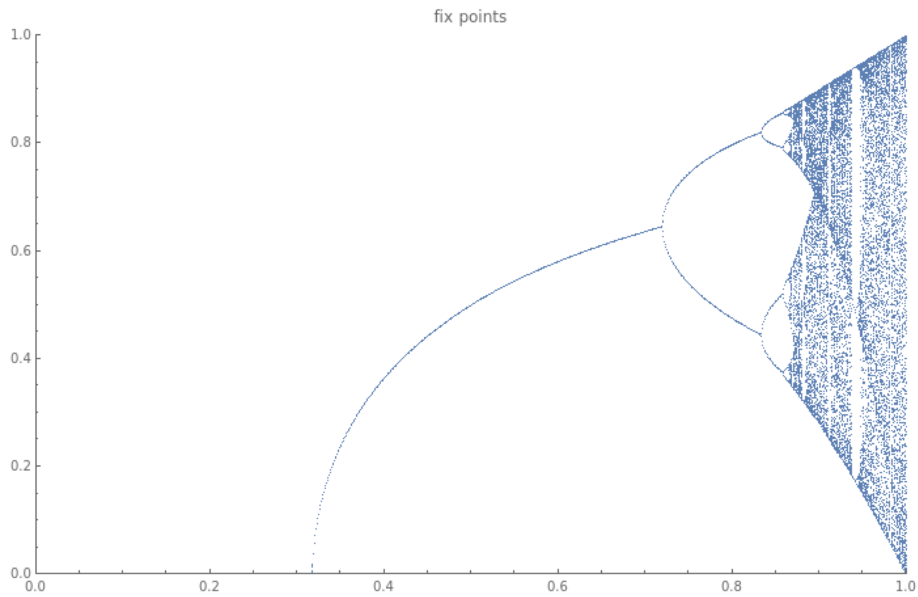


Figure 2.1: Fix points of $g(x) = asin(\pi x)$

By applying Mathmatica Desktop version, the first three doubling periods' coordinates are identified as below:


Figure 2.2: The *a* values' first three doubling periods

According to the calculation formula of the first Feigenbaum constant, the constant is the limiting ratio of each bifurcation interval to the next between every period doubling [Bri91]. That is:

$$FeigenbaumConstant(1st) = \lim_{n\to\infty} \frac{a_{n-1} - a_{n-2}}{a_n - a_{n-1}} = 4.669201609 \tag{2.1}$$

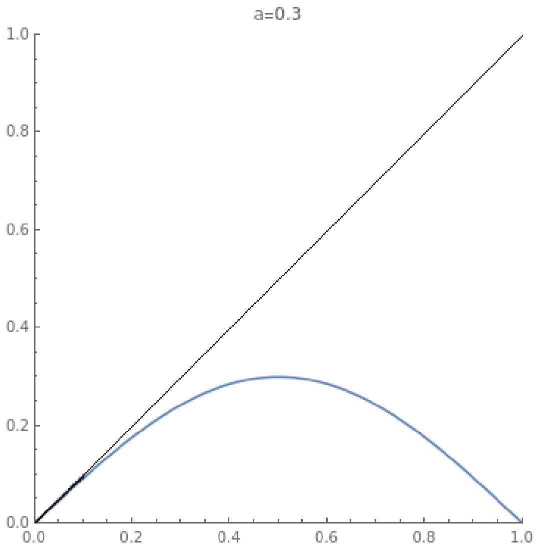


Figure 2.3: a = 0.3

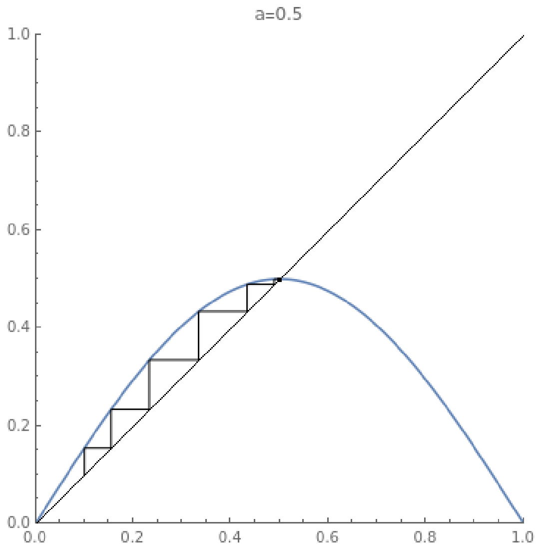


Figure 2.4: a = 0.5

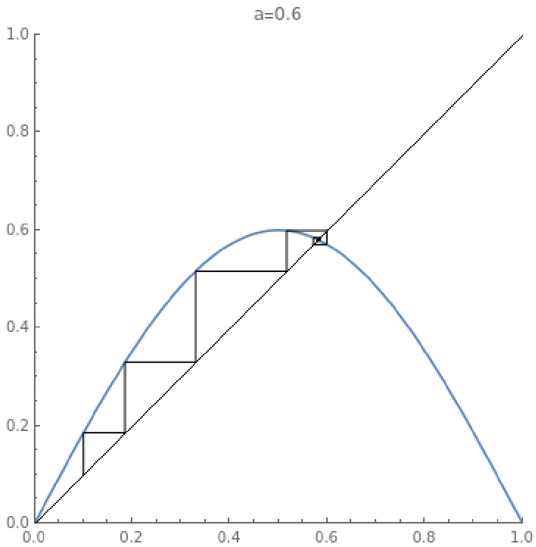


Figure 2.5: a = 0.6

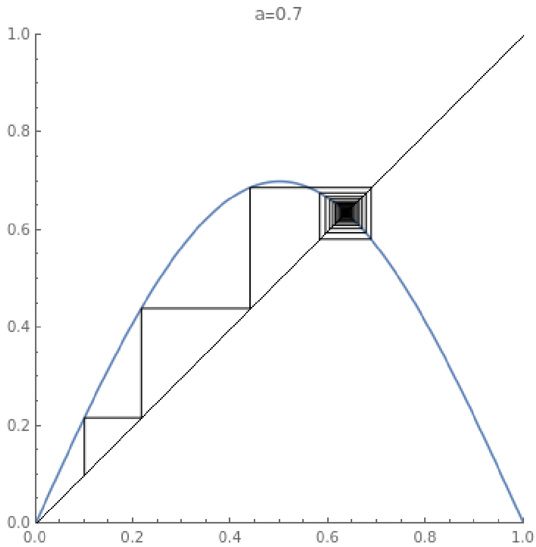


Figure 2.6: a = 0.7

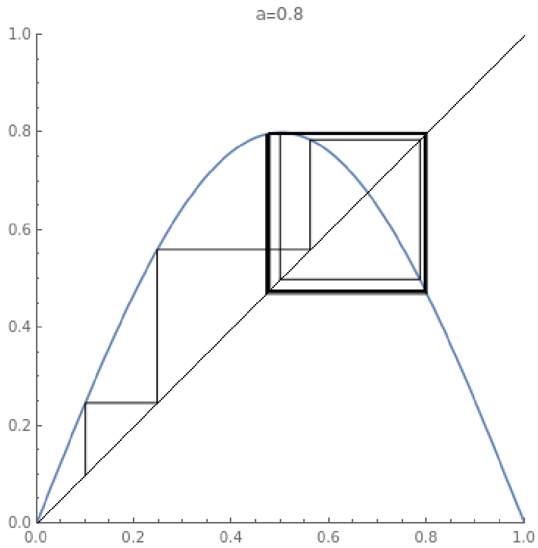


Figure 2.7: a = 0.8

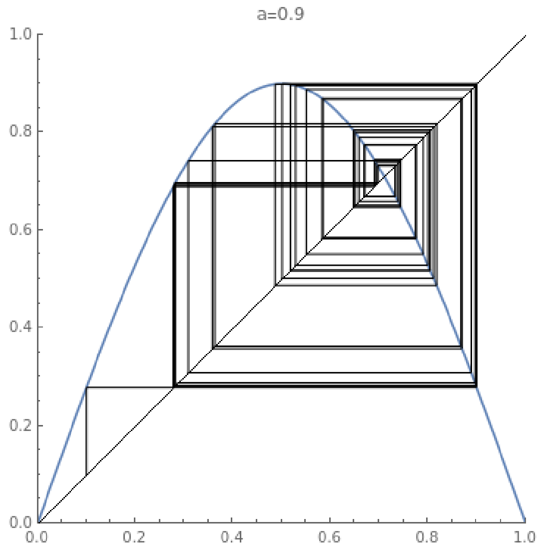


Figure 2.8: a = 0.9

As we can see from the above diagrams, different $a$ values are tested in order to confirm the the patterns of fix points which are found in Figure 2.1. The first doubling period happens when $a = 0.719067$ that can be verified by Figure 2.5 and 2.6 which show the fix points. When $0.71 < a < 0.83$, it can be observed that the $a$ value loses its stability (see Figure 2.7). When $a = 0.9$ (Figure 2.8), the pattern of instability is even larger than the previous diagram. These results are coherent with the findings of Figure 2.1, Figure 2.2 and the development pattern of logistic map when the value of $a$ increases, hence, the previous findings are verified.

Along with the calculation formula of Feigenbaum constant (Formula 2.1) and observed $a$ values from the first three doubling periods (Figure 2.2), the Feigenbaum constant can be calculated. Due to the limitation of observing Figure 2.1, only three doubling periods can be retrieved from the diagram, a lack of precision of calculation could remain in the final results which can be considered as the limitation of this experiment. The estimation is proposed to be conducted by utilizing the observed values. In order to ensure the calculation accuracy, the $a$ value will be kept as six decimal places. The estimation process is as following:

$$a = \frac{0.832922 - 0.719067}{0.858757 - 0.832922} = 4.485306 \tag{2.2}$$

Therefore the estimation of Feigenbaum constant based on first three doubling period is 4.485306 in this examination.

# 3 Cellular Automata 1D

The task regarding one dimensional cellular automata required that rule 225 for a seed of length 1000 should be run. All cells are required to start off as white with the exception of the tenth one which should be a black cell. After that has been done, then it should be iterated

1000 times and the figure should be presented along with the rule number, which was 225, to be written in base 2 and explain the rule.

To solve the task, the template for one dimensional cellular automata provided in the course's material was used. The code in question was slightly modified to suit the requirements of the task and is the one below:

```
dim=225
seed1 =Table[0,{dim}]
seed1[[10]] = 1
BaseForm[225,2]
RulePlot[CellularAutomaton[225]]
ArrayPlot[CellularAutomaton[225, seed1, 1000], Mesh -> True]
```

To explain the code, it simply means that a seed is generated with the rule in question however a black cell is to placed in position 10 and this should be iterated 1000 times.

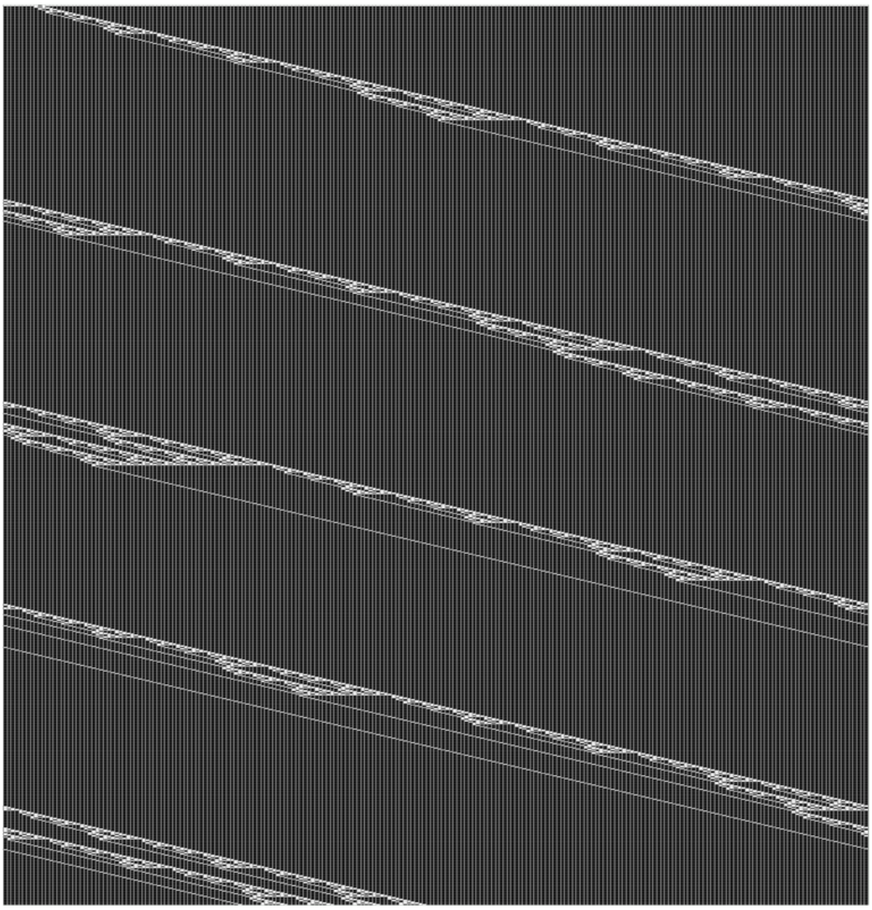

Figure 3.1: Code plotted with Mathematica

Rule 225 in form of base 2 is 11100001. Considering that in the first generation all of the cells are white (W or 0) except for the one in the tenth position which is black (B or 1), then the question arises: "How do we compute the second generation?" To go from here, the neighbourhood (which in this case, it will be the nearest 3 adjacent cells) is of essence so that at the very least, the state of the tenth position can be determined. The next question would be: "In how many different ways can that neighbourhood be configured?", which is where the rule itself comes in. Rule 225 states that if the pattern is BBB (111) then the new state for the center cell is going to be black as well (1). If the pattern is BBW (110) then the new state for the center cell is going to be B (1). The BWB pattern will lead to a center cell that is B (1). The BWW pattern will lead to a center cell that is W (0), The WBB (011) pattern will give a center cell that is W (0). The WBW (010) will give a W (0), WWB (001) will equal to a W (0) and lastly WWW (000) will give a black (1) center cell. The rule therefore defines the configurations so for the next neighbourhoods.

# 4 Cellular Automata 2D

The task regarding two dimensional cellular automata requires that the rule integer 90016 is run and should start with a wall, that is a square of black cells. The grid should be 100 by 100 and the wall should be placed at rows 10 and 90. Afterwards, it should be iterated 1, 2, 3,... 10, 100, 500 times and the findings should be shown along with an explanation to the rule integer itself which should be expressed in base 2. That is to be done with the intention of explaining when birth and survival will happen.

The task was first solved by using the following code:

```
Clear["‘*"]

WalledCities = {90016, {2, {{2, 2, 2}, {2, 1, 2}, {2, 2, 2}}}, {1, 1}};
seed = Table[0, {100}, {100}];
seed[[10 ;; 90, 10]] = 1;
seed[[10 ;; 90, 90]] = 1;
seed[[10, 10 ;; 90]] = 1;
seed[[90, 10 ;; 90]] = 1;

BaseForm[90016,2];

ArrayPlot[CellularAutomaton[WalledCities, seed, {{{0}}}], Mesh -> True]
ArrayPlot[CellularAutomaton[WalledCities, seed, {{{1}}}], Mesh -> True]
ArrayPlot[CellularAutomaton[WalledCities, seed, {{{2}}}], Mesh -> True]
ArrayPlot[CellularAutomaton[WalledCities, seed, {{{3}}}], Mesh -> True]
ArrayPlot[CellularAutomaton[WalledCities, seed, {{{100}}}], Mesh -> True]

Animate[ArrayPlot[CellularAutomaton[WalledCities, seed,
{{{n}}}]], {n, 0, 100, 1}, AnimationRate -> 20,
AnimationRunning -> False]
```

To explain the code, in reference to the Wolfram documentation convention the WalledCities defines the rule that is to be used, in this case 90016 and then the lone 2 is the number of states that a cell can inhabit (black or white). The rule form that it takes is "n,k,wt1,wt2,...,rspec" [Res17]. The 1,1 is the so called rspec in which it specifies how each cell depends on the neighbourhood next to it and so forth. The matrix specified in the rule form are how the cells are weighted. The seed is starting off with a table of size 100 by 100 and then the wall of black cells is placed on it. Finally, BaseForm is used to give the rule integer 90016 in base 2 to aid the understanding of it.

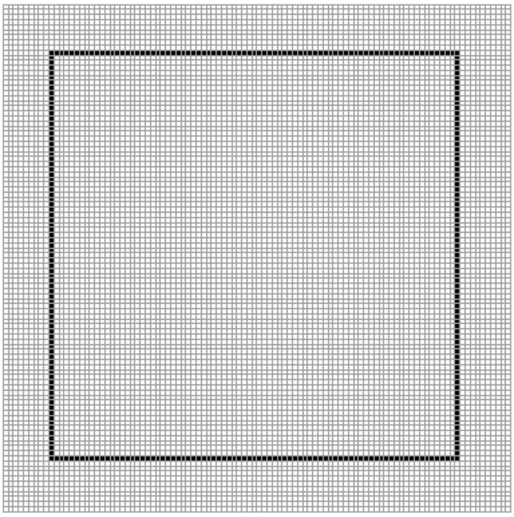

Figure 4.1: The wall of black cells

The iterations begin and these are the results from 1 to 3:

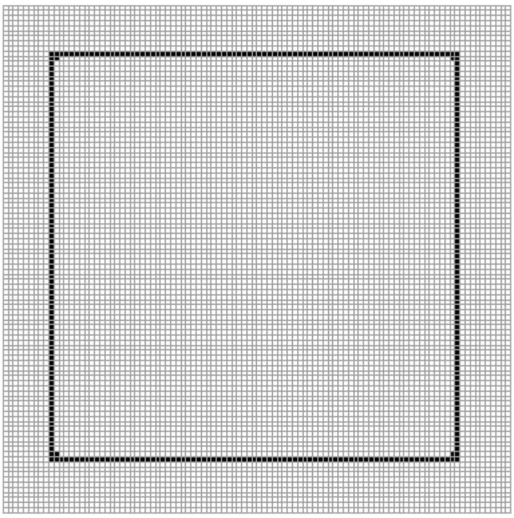

Figure 4.2: 1 iteration

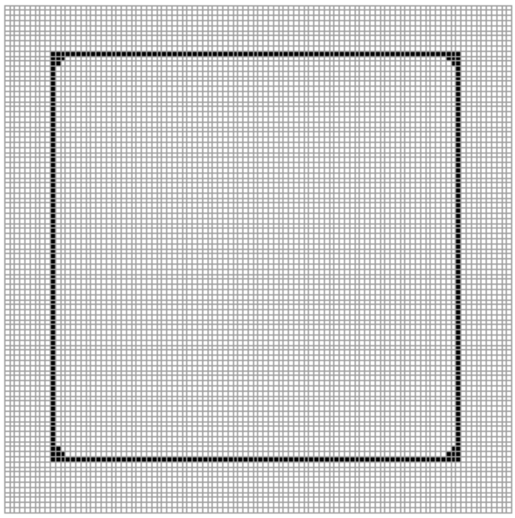

Figure 4.3: 2 iterations

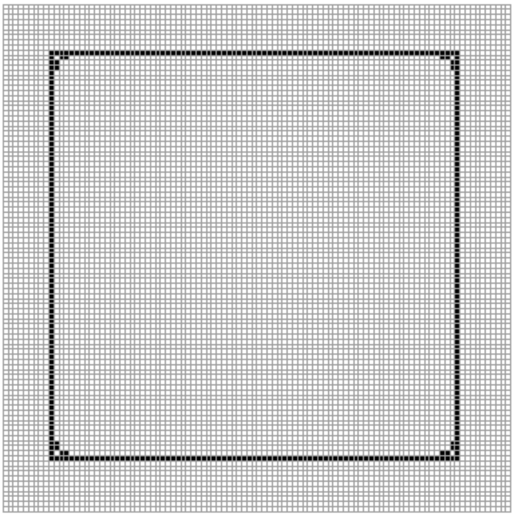

Figure 4.4: 3 iterations

After having finished the three iterations, for the next ones, the values 10, 100, 300, 500 and 1000 were chosen.

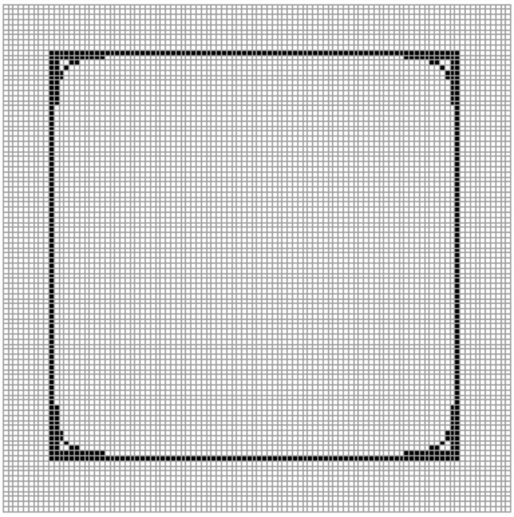

Figure 4.5: 10 iterations

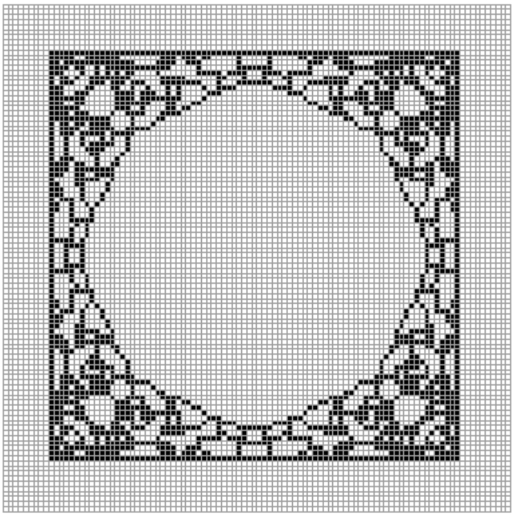

Figure 4.6: 100 iterations

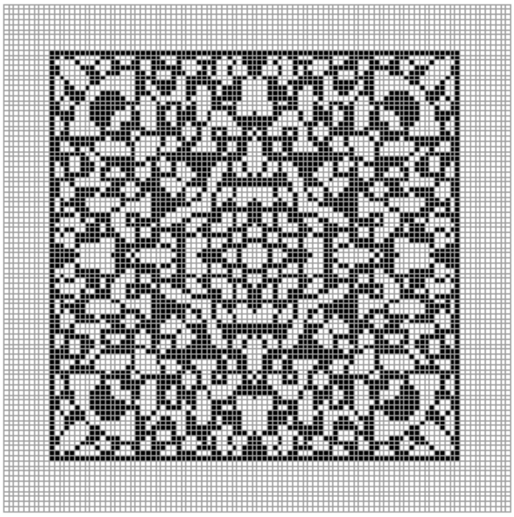

Figure 4.7: 300 iterations

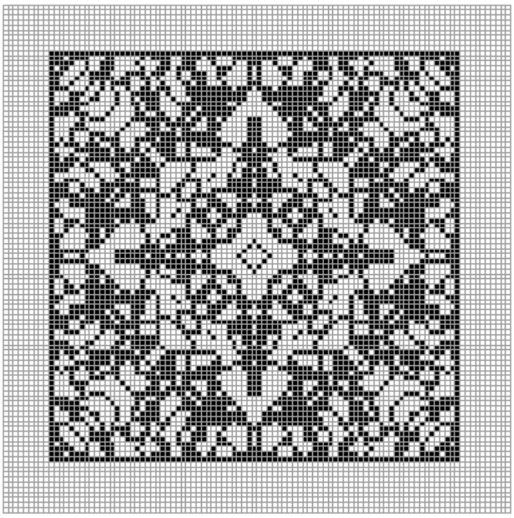

Figure 4.8: 500 iterations

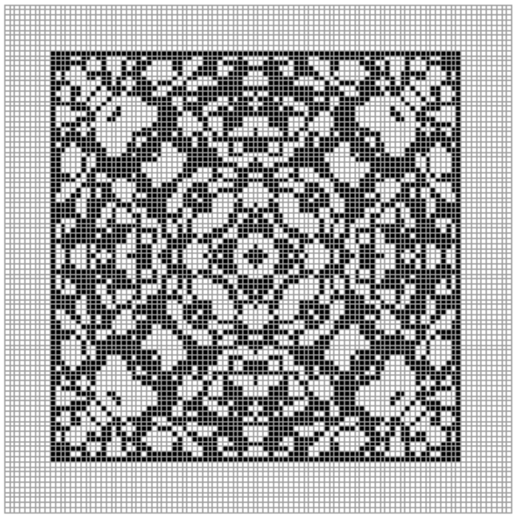

Figure 4.9: 1000 iterations

After looking at the figures, it can be noted that the wall itself does not expand however as the number of iterations grows, the actions within the wall increase. This is a characteristic of the rule in the "Life" family called walled cities. Using Wolfram's 2D Cellular Automaton Animations application and experimenting with different birth, survival values, it was discovered that for the rule integer 90016 the birth, survival values are B45678/S2345 which is in tandem with 90016 in base 2 that is 10101111110100000 [Bog11].

| N = black cells | M = state of middle cell | 2N + M | $2^{(2N+M)}$ | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | |
| 0 | 1 | 1 | 2 | 0 | |
| 1 | 0 | 2 | 4 | 0 | |
| 1 | 1 | 3 | 8 | 0 | |
| 2 | 0 | 4 | 16 | 0 | |
| 2 | 1 | 5 | 32 | 1 | X |
| 3 | 0 | 6 | 64 | 0 | |
| 3 | 1 | 7 | 128 | 1 | X |
| 4 | 0 | 8 | 256 | 1 | X |
| 4 | 1 | 9 | 512 | 1 | X |
| 5 | 0 | 10 | 1024 | 1 | X |
| 5 | 1 | 11 | 2048 | 1 | X |
| 6 | 0 | 12 | 4096 | 1 | X |
| 6 | 1 | 13 | 8192 | 0 | |
| 7 | 0 | 14 | 16384 | 1 | X |
| 7 | 1 | 15 | 32768 | 0 | |
| 8 | 0 | 16 | 65536 | 1 | X |
| 8 | 1 | 17 | 131072 | | |

Table 4.1: Workings of the rule integer 90016.

| Neighbours | Survival | Birth |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 2 | 1 | 0 |
| 3 | 1 | 0 |
| 4 | 1 | 1 |
| 5 | 1 | 1 |
| 6 | 0 | 1 |
| 7 | 0 | 1 |
| 8 | 0 | 1 |

Table 4.2: Workings of the birth and survival of 90016.

The rule states that if a cell is white however surrounded by 4, 5, 6, 7 or 8 black cells then it will go from white to black and if a cell is black although surrounded by 2, 3, 4 or 5 black cells, it will survive.

# References


[Bri91] Keith Briggs. "A Precise Calculation of the Feigenbaum Constants". eng. In: *Mathematics of computation* 57.195 (1991), pp. 435–439. ISSN: 0025-5718.

[Bog11] Kovas Boguta. *2D Cellular Automaton Animations*. Accessed:28-April-2021. 2011. URL: `http://demonstrations.wolfram.com/2DCellularAutomatonAnimations/`.

[Res17] Wolfram Research. *CellularAutomaton*. Accessed:28-April-2021. 2017. URL: `https://reference.wolfram.com/language/ref/CellularAutomaton.html`.