NaCo 23/24 assignment 1 report

Aloysius Andries Arthur Maria, Aloysius; van den Berg, Aquilla; Oosterhoff, Marloes

Group number *A*15

Abstract. The following report describes the basic features of a Cellular Automata and its usage in urban planning [8]. It also presents a basic programmed model showing the different types of Wolfram classes in various CAs and how to determine a Wolfram class based on two different discrete measures.

1 Introduction

There are various methods for generating new algorithms and models. In natural computing we take inspiration from nature, to then apply it to various computational models. Cellular automata are no exception to this principle. Inspired by the growth of crystals and wanting to model self-reproducing organisms, the basic workings were developed by John von Neumann. The name "Cellular Automata" was coined by von Neumann in 1966, it was popularized, however, by Stephen Wolfram in 1983 [9].

A cellular automaton (CA) is a discrete model studied in various disciplines, including but not limited to: mathematics, theoretical biology and computability theory [7]. It consists of a grid of cells where for each discrete time step a new row is generated. The generation of every cell in the next time step is based on the neighborhood of the previous time step. According to the binary structure of the CA rule the outcome per combination of neighborhood cells is determined.

2 Problem Description

Perhaps the simplest type of CA is the one-dimensional CA, where we have determined the neighborhood to contain the cell and one direct neighbor on either side. CAs can be displayed in various ways; we have opted for a static two-dimensional plot, but they can also be presented dynamically through animations [3].

The results of a cellular automaton can be categorized in one of four qualitative behavioral classes: the Wolfram classes. These classes have also been developed by Wolfram in 1983. A CA's Wolfram class is determined by the way the CA develops through time: its behavior. Therefore a good measure for determining these classes is the amount of cells changed per discrete time step. Another measure for CA behavior is the amount of cells belonging to the same type per discrete time step. We will explain how these measures can be implemented below.

The classes are described by Ilachinski (2001) [6] as follows:

The first class involves CAs which evolution leads to a homogeneous state, in which all cells eventually attain the same value. In terms of the chosen measures, we would've reached this class when the number of changed cells per discrete time step has become 0 and the amount of zeros per time step is equal to the array size of the CA.

The second class is described as the CAs' evolution leading to either simple static states or periodic and separated structures. We can find this behavior back in the measures as a regularly changing amount of changed cells per time step, as well as a periodic change in amount of zero-cells per time step.

The third class consists of chaotic CAs with non-periodic patterns. This behavior can be found in the measures by randomly changing amounts of changed cells, as well as seemingly unconsistently changing zero-cells per timestep.

The fourth class is more special, as it consists of complex, localized propagated structures. This class is more difficult to find back in the measures, but is visually more recognizable. The amount of changing cells will change here in a algorithmic manner without repeating itself consistently. This will similarly occur in the number of zero-cells per discrete time step. This class is the most informative of all the classes.

3 Results

In Python 3.11 we implemented a program using the packages NumPy [4] and Matplotlib [5]. Per CA, the program outputs a visualization plot of the CA, with an array size of 60, and presents two measures per time step in the console. These measures are: the amount of cells changed and the amount of zero cells (including the boundary zeros, which are static).

Three different starting states were used: one where the centre cell starts out as one; the other where there is a single "one-cell" at the halfway and third-way point, as well as three consecutive ones at the fifth-way mark. The final starting state is randomly generated.

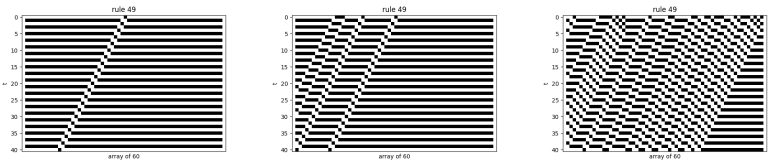

Fig. 1: Rule 49 with different starting states.

Figure 1 shows rule 49 starting from different starting states. Although the image is slightly different for every starting state, the overall behavior remains the same: predictably changing. Therefore this rule fits into Wolfram class **two**.

Whether starting state determines the outcome of the CA depends entirely on the rule, sometimes the effects are larger than other times.

Figure 2 shows rule 110 with different starting states. The starting states have a larger effect on the outcome than for rule 49. The behavior is less predictable but shows some kind of interaction and regularity. There is a distinction between 'background noise' and a more overarching shapes, which are preserved. This rule belongs to Wolfram class **four**. For Wolfram class four, you often need to add more time steps in order to see more meaningful patterns.

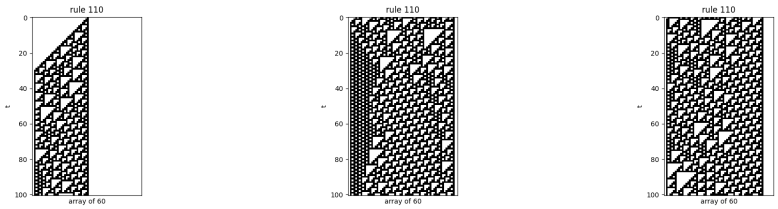

Fig. 2: Rule 110 with different starting states

Choosing a more complex starting state results in more complex shapes at times, this is especially visible in rule 120 below. Having a starting state where more consecutive cells are one results here in thicker structures.

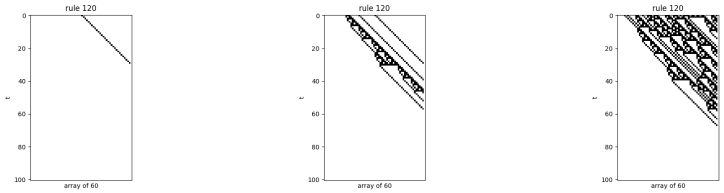

Fig. 3: Rule 120 with different starting states.

Regardless of the varying starting states, rule 120 still ends up uniformly zero. However, we can now see the effect of adding more ones to the starting state: the automaton takes longer to reach homogeneity. As the automaton doesn't develop after a certain time step, we can put this rule in Wolfram class **one**.

Sometimes adding consecutive ones to the starting structure does not result in a thicker structure, we can see that with rule 165.

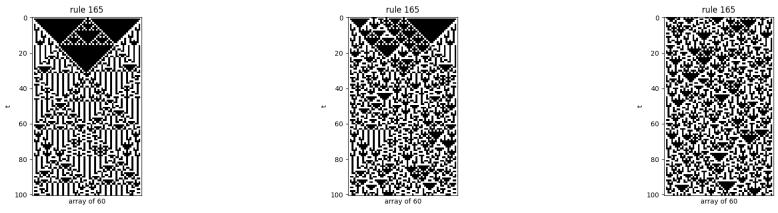

Fig. 4: Rule 165 with different starting states.

Rule 165 is an example of a Wolfram class **three** structure. The structure is entirely unpredictable and there is no real pattern to be found. Comparing this to Wolfram class four structure rule 110, we can see that although this automaton starts more structured, it chaotically ”explodes” into irregular structures, which cannot be said for rule 110. The shapes in rule 165 are not as preserved either, but seem to be ’collapsing’.

4 Application of CA in practice

In the paper of ‘A cellular Automata (CA) model based on irregular cells: application to small urban areas’ by Nuno Pinto and Antonio Antunes, they attempt to use irregular cell (instead of regular cells) to more accurately depict and predict land use changes for developing small sized urban areas [8]. As more urban areas are experiencing sharp growth problems start to crop up in the planning process, these issues are based on past failures but also to prepare for more contemporary challenges within land-use policy have in tackling urgent modern needs and ensuring a proper transition to a sustainable future. To be able to elect the best probable outcome, it is vital that such planners and decision makers can forecast how their decision can effect urban environments over time and effect the whole spatial ecosystem.

CA has a strong history of being utilized within the field of urban planning, more specifically models were created to simulate the process land use changes in different regions. These models were applied to real life urban data and can enhance the understanding of urban dynamics and used CA models to simulate urban growth. The accuracy of these models are usually compared to reference maps of actual changes to gauge their accuracy. As time developed and more breakthroughs and additions to CA model’s potential have been realized in this field, more variables and factors were of the models were expanded on in order to achieve higher accuracy/agreeability. Although hypothetical areas are where the models might be tinkered with, these models are ultimately to be applied in order to forecast real life urban growth scenarios [1]. The main aim of this paper is to improve the simulation of urban change by enhancing the predictability of CA models in relation to the underlying spatial structure in small urban areas.

Certain spatial issues with applying CA to stimulate urban growth can be solved with the use of irregular cells as it allows urban form issues to be dealt with more efficiently. The irregular cells are represented by census blocks (statistical areas contained and bounded by geographical features such as roads, rivers, woods, property lines, districts etc) with large variety in sizes. All in all these blocks are an improvement from models that use regular cells based obtained from satellite images that use pixels to represent the spatial structure of the territory that can be limited based on the quality of the image resolution. Furthermore these regular cell models of urban planning mostly only hold information of the designated land use and not much of other valuable information. This very systematic breakdown of geographical land areas through irregular cell models allows to accommodate the more complicated reality of modern urban planning, and hopefully it will give us a more realistic and up to date representation of how urban development elapses throughout the years. The model functions based on constraint of land use demand (based on a particle swarm algorithm suitable for dealing with a large number of parameters) associated with each census block (cell). Additionally vast amounts of reliable demographic and socioeconomic data can be used in the application of the model through these land constraints.

The specifications in the CA model is as follows:

The model operates on a 2 dimensional irregular grid represented with cellular fabric of different sizes obtained and based on preexisting census blocks. At its most basic form irregular cellular automation is a type of CA with no predetermined restriction on the grid structure of its cells. The cells will artificially stimulate the structured and reliable socioeconomic and infrastructural information with a form that reliable holds true to the actual urban form of areas (based on streets, roads, natural barriers etc). This paper outlines 6 states that each cell can hold that represent a land class (different land uses): urban low density (UL), urban high density (UH); industry (I), urban expansion (XU), industrial expansion XI), and highly restricted uses (R). Neighbourhoods are also done slightly different, they elected to utilize circular neighbourhoods in their model. Moreover the number of cells in a neighbourhood aren't predetermined by a set distance or radius rather it implements a variable circular neighbourhood where the radius is a calibrated parameter whose value is influence by all other parameters of the model In this manner the model is more realistic and can more accurately model land-use interaction due to the volatility of neighbourhood size and density perception in different urban areas.

The transition rules within their model incorporates the following factors: planning regulations, land-use suitability, and neighbouring interactions. Ultimately the rules are expressed through a potential that reflects the propensity/capacity of each cell to change the states based on the weighted value of each of the previously mentioned factors. The full expression is as follows (but I won't fully explain the equation, but the three factors are represented by the S, A and N and have their own coefficients):

$$P_{i,s} = (v_P S_{i,s} + \chi_P A_i + \theta_P N_{i,s})\xi, \qquad \forall\, i \in \boldsymbol{C}, \quad s \in \boldsymbol{S},$$

The time steps being taken is also variable, usually only a 1 year time step is customary and sufficient however the model can use time steps of 1, 5 or 10 years. The decision to use a time-step is based on the users context and case that they wish to model, this can be influenced by a plethora of factors such as data availability or the expected rate of change for a cell state to change based on the planning environment (population/demographic/economic expectations). However in the case study they used a timestep of 10 years. The way the model deals with land-use demands also differs from traditional methods due to the different areas of cells with corresponding differences in population, economics and building density. In order to simulate the observed value of land consumption for every cell state the model must allocate population or employment based on given threshold of their densities.

The model had to be calibrated to fine tune the possible adjustment between the simulated outcome and the reality that is being modelled. Out of the two approaches outlined in the paper (sensitivity test using visual comparison and optimisation feature using a fitness function) the optimisation approach was chosen as this approach performed better with higher calibration parameters compared with the sensitivity test. The fitness function that was chosen was based on a kappa index fitness function that was modified to account for distortion of cell states that cannot take part in urban dynamics (state R) by not including it in the computation as the inclusion of these cells would produce large but meaningless agreement between the simulation and the reference. The values this kappa index can take is in the range of 1 and 0 where 1 is complete agreement and 0 meaning no agreement or independence [10]. The model was calibrated using datasets from 1991 and 2001. A group of 60 particles of 48 parameters were randomly generated considering plausible ranges for each parameter and utilized for individual CA runs for each

particle swarm generated. The condition of halting the algorithm was when the fitness function did not improve more than 0.1 percentage. The calibration results were successful in many aspects, mostly in approximately identifying areas where change occurred. The individual prime parameters also yielded agreeable/promising results that I don't have the time to divulge into. The final fitness function of the modified index had a value of 0.621 compared to a 0.774 value of the basic function (overrate due to R), but the agreement is adequately high enough and are closed to global agreement attained for the case study (0.771). Yet the number of cells whose transition was exactly determined (accuracy and exploitation lacking) by the simulation was lacking considering the high fitness value urban land use transition had prediction accuracy was 19/94 cells whose simulated transition matched reality. These problems mainly persist in large cells inside urban centres is where these big differences in agreement lie due to a myriad of problems with the model and calibration decisions.

With our limited knowledge on urban planning, I do believe that CA was the right approach to tackle this question considering the rich history of this methods application in the field of urban growth/planning, it has been adopted since the 1980s [1]. Building on the shoulders of giants while also attempting to improve on it based on newer techniques (irregular cell blocks, particle swarm algorithms) to better simulated realistic urban change is a step in the right direction to better inform planners of realistic what if scenarios. There are other options out there to simulate land use-land cover change such as Partial differential equations, Markov Chain models and artificial neural networks models [2]. Nevertheless CA is still the most commonly used one at the time of publishing. So considering they were building on these CA models and trying to improve on it, it was only logical they used CA in their situation. On top of that the nature of the CA model to consider the spatiotemporal dynamics of urban growth makes it more advantageous than the other existing models.

There are certain aspects to improve from this paper. There were biases based on land-use demand as certain large cells attract more development than the simulation predicts. They could solve this by divvying up the larger areas into smaller ones from the start. There are also different manners in which they can improve the cell transition potential that can include more varied modes of transportation to more accurately predict cell state change as mentioned in their transition rules. They could also build on this current model by implementing a multiscale approach as urban phenomena involved many spatial facets which should also be simulated. Personally I would also attempt to attempt to use a similar model in a more exploitative approach as the number of iterations are quite small. The factors of land-use demand is also much too simplistic as many factors are not considered, however three factors are not enough and others should be added to better model demnad

5 Conclusion

In this paper we have discussed a few ways in which we can determine the Wolfram class of a cellular automaton. Using a Python script we were able to display CAs of various time step lengths and starting phases. Cellular automata are used in various disciplines. Particular attention was raised to a paper by Pinto and Antunes regarding the usage of CAs in urban development and how irregular cell usage and particle swarm algorithm application leads to more agreeable CA models.

References

1. Chakraborty, A., Sikder, S., Omrani, H., Teller, J.: Cellular automata in modeling and predicting urban densification: Revisiting the literature since 1971. Land **11** (2022). https://doi.org/10.3390/land11071113, https://doi.org/10.3390/land11071113
2. Chen, Z., Huang, M., Zhu, D., Altan, O.: Integrating remote sensing and a markov-flus model to simulate future land use changes in hokkaido, japan. Remote sensing **13** (2021). https://doi.org/doi.org/10.1068/b36033, https://doi.org/10.3390/rs13132621
3. Floreano, D., Mattiussi, C.: Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies. The MIT Press (2023)
4. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E.: Array programming with NumPy. Nature **585**(7825), 357–362 (Sep 2020). https://doi.org/10.1038/s41586-020-2649-2, https://doi.org/10.1038/s41586-020-2649-2
5. Hunter, J.D.: Matplotlib: A 2d graphics environment. Computing in Science & Engineering **9**(3), 90–95 (2007). https://doi.org/10.1109/MCSE.2007.55
6. Ilachinski, A.: Cellular Automata: A Discrete Universe. World Scientific (2001)
7. Li, T.M.: Cellular Automata. Nova (2011)
8. Norte Pinto, N., Pais Antunes, A.: A cellular automata model based on irregular cells: Application to small urban areas. Environment and Planning B: Urban Analytics and City Science **37** (2010)
9. Sarkar, P.: A brief history of cellular automata. ACM Computing Surveys **32**(1), 80–107 (2000). https://doi.org/10.1145/349194.349202, https://doi.org/10.1145/349194.349202
10. Yu, C.H.: Test–retest reliability. Encyclopedia of Social Measurement pp. 777–784 (2005). https://doi.org/10.1016/b0-12-369398-5/00094-3, https://doi.org/10.1016/b0-12-369398-5/00094-3