CABuilder: Graphical Representation and Pseudorandom Sequences from Cellular Automata*

Nicoli Pinheiro de Araújo, Elloá B. Guedes

[1]Núcleo de Computação
Escola Superior de Tecnologia
Universidade do Estado do Amazonas
Av. Darcy Vargas, 1200 – Manaus – Amazonas

npda.eng@uea.edu.br, ebgcosta@uea.edu.br

Abstract. *Cellular automata are a computational model based on cells which evolve based on their own values and on the values of their neighbors. They are considered simple under the mathematical perspective, but despite that they are capable to generate complex behaviors. Several applications of such computational model include from chemistry reactions to fractals. Considering its importance, this works aims at showing the* Cellular Automata Builder *(*CABuilder*), a software tool built in Python which produces graphical representations and pseudorandom sequences from two variants of cellular automata. This software is free, open source and is recommended for researchers in cellular automata; for teachers and students interested in this computational model; and also for developers who demand pseudorandom sequences in their applications, such as in simulations, cryptography, among others.*

1. Introduction

Theory of Computation is an area from Computer Science which aims at studying the capabilities and limitations of computers. Instead of considering a particular device, this area uses *computational models* which model real world devices but abstracts away its material details to make calculation [Fernández 2009].

On of the most simple computational models is the *finite automata* which started out the *Automata Theory* research area. The main concepts of such theory were introduced in the 50's as a result of many efforts from several researchers including mathematicians, linguists, neurophysiologists, electrical engineers, among others. The seminal works aimed at the development of machines capable of modeling the cognitive processes of the human brain. On 60's and 70's many important theoretical progress has been made on this area [Perrin 1995]. In the last two decades, development of new automata models and applications have called attention. Thanks to that, Automata Theory has been considered important not only in the academic field, but has been extensively used in real world applications. Recently, new automata variants include probabilistic and quantum elements [Guedes and Lula Jr. 2010]

The *finite cellular automata* are a variant from the canonical definition of automata. They were proposed in 1950 by von Neumann and Ulam [Sarkar 2000] and were extensively studied by Wolfram [Wolfram 1984]. They are structurally simple yet capable to generate a complex behavior. Applications of this computational model include biological systems, eavesdrop detection, image processing, fractals, among others [Schiff 2010]. Thanks to their complex behavior, a remarkable application of cellular automata is in the generation of pseudorandom sequences [Gentle 2003, Wolfram 1986].

*Concluded Work

Considering the importance and the applications of cellular automata, this article aims at introducing a software tool called *Cellular Automata Builder* (CABuilder, for short). This free and open source software was programmed in Python and is capable to generate graphical representations of elementar and totalistic cellular automata. Besides, it can generate binary pseudorandom sequences from these automata. Such sequences can be used, for instance, in simulations and probabilistic algorithms.

The motivation to create this software started when the authors of this work needed to analyze pseudorandom sequences from cellular automata but the available alternatives were not free or demanded much effort on configuration. So, having in mind that it needed to be easy to use, but with a set of features to create, visualize and produce data from cellular automata, the CABuilder was conceived. Considering such characteristics, this software is well suited for researchers in cellular automata; for teachers and students interested in this computational model; and also for developers who demand pseudorandom sequences in their applications.

To present CABuilder, this paper is organized as follows. A brief historic contextualization and the concepts of cellular automata can be found in Section 2. An overview of CABuilder – including its functionalities and details of implementation – is presented in Section 3. Related work is discussed in Section 4. Lastly, final remarks and future work are shown in Section 5

2. Cellular Automata

Cellular Automata are a computational model that represents many natural systems. It is strongly based on Mathematics, but with a very simple definition. Such kind of automata is composed of *sites* which are identic, discrete and that can assume values from a finite set. The value of each site is modified in discrete time steps according to a deterministic rule which depend on their own value and on the value of their neighbors, characterizing the *evolution* of a cellular automaton [de Moraes 2007].

The seminal work regarding cellular automata was proposed in 1950 by John von Neumann e Stanislaw Ulam. They developed abstract models of cellular structures capable of self-reproduction using concepts from electronic circuits. The main idea behind their proposition was that complex behavior of computer and other devices could be generated by complex rules [Sarkar 2000]. From their work, it was possible to derive a formal definition for such automata, as shown in Def. 1, which allowed the development of several variants of this model.

Definition 1 (**Celular Automata [Wolfram 1984]**). An unidimensional cellular automaton is composed by a line of sites $\{a_1^{(0)}, a_2^{(0)}, \ldots, a_n^{(t)}\}$ and is defined by three parameters (r, k, ϕ). The value $a_i^{(t)}$ of the site i in the time t is obtained according to the following expression:

$$a_i^{(t+1)} = \phi \left[a_{i-r}^{(t)}, a_{i-r+1}^{(t)}, \ldots, a_{i+r}^{(t)} \right], \tag{1}$$

where k is the number of the possible values each site can assume, in the range $[0, k[$; r is the amount of neighbors sites that must be taken into account to determine the value of a site; and ϕ is a deterministic rule.

There are several variants of cellular automata in the literature. They differ according to the parameters specification. The *elementar cellular automata*, for instance, have

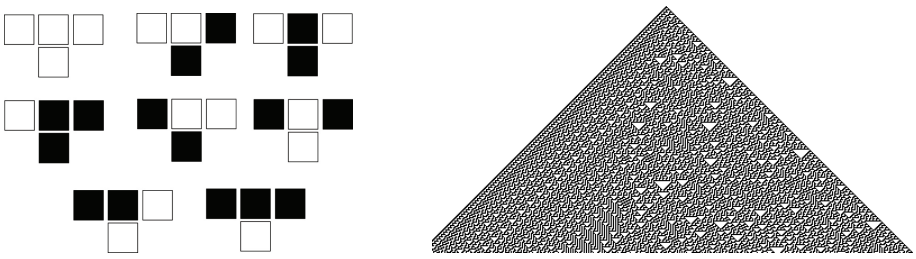
$k = 2$, $r = 1$ (immediate neighborhood) and rule ϕ based on the decomposition of a number into binary (rule number) [Wolfram 2002]. The decomposition of a number into binary means writing this number for the decimal basis to the binary basis. Each resulting bit is associated to one of the results of the rule ϕ.

One of the most known elementar celular automaton is called *Rule 30* [Wolfram 2002, Wolfram 1984]. This automaton rule is defined as follows:

$$b_i^{(t+1)} = \left(b_{i-1}^{(t)} + b_j^{(t)} + b_{i+1}^{(t)} + b_i^{(t)} \cdot b_{i+1}^{(t)}\right) \mod 2. \quad (2)$$

A better visualization of such automaton rule can be found in Fig. 1a. It is interesting to notice that this automaton has only the colors black and white because $k = 2$; the value of the parameter r is 3 because its state in a next step of time depends only on its own state and of his left and right neighbors' state, as shown in Eq. 2.

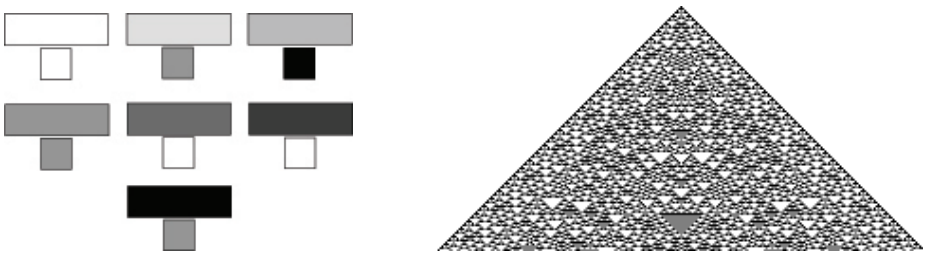
Considering the initialization of such automaton with a single site with value 1 (black site) and observing its evolution according to the rule in Eq. (2), the result after 200 iterations is shown in Figure 1b. Despite the simple rule, the resulting pattern looks complex.



(a) Rule **(b) Evolution**

Figure 1: Example of elementar celular automaton "Rule 30".

Another variant of celular automata is based on *totalistic rules*. According to this kind of automata, the value that a site assume is based on the average of its own value and of its neighbors. To this kind of cellular automata, in particular, besides the colors black and white, shades of grey can emerge in the final pattern. The rule 777 depicted in Fig. 2a is used to illustrate the totalistic cellular automata.



(a) Rule **(b) Evolution**

Figure 2: Example of totalistic celular automaton "Rule 777".

It is interesting no notice that since $k > 2$, the shades of grey are used to represent the values besides 0 and 1. It happens because for each given k, there are $3 \cdot k - 2$ possible

colors. Taking all the possible combinations, there are $k^{3 \cdot k-2}$ cellular automata for each k.

Considering the cellular automata definition presented and both variants introduced, Wolfram verified that four different patterns can be generated by such automata: (i) patterns that disappear along the time; (ii) patterns that evolve in a fixed size; (iii) patterns that grow indefinitely in a fixed speed; and (iii) patterns that grow and diminish in an irregular manner [Wolfram 1984]. All the patterns identified are shown in Fig. 3. The cellular automaton in Fig. 3a is elementar with rule number 0. This automaton disappears after the second step of evolution. The cellular automaton in Fig. 3b is also elementar, but with rule number 12. It repeats the initial pattern along the time. The automaton in Fig. 3c is the rule 30 shown previously, whose pattern grows along the time. Lastly, the automaton in Fig. 3d has totalistic rule 600 with $k = 3$, illustrating a pattern of the last type.

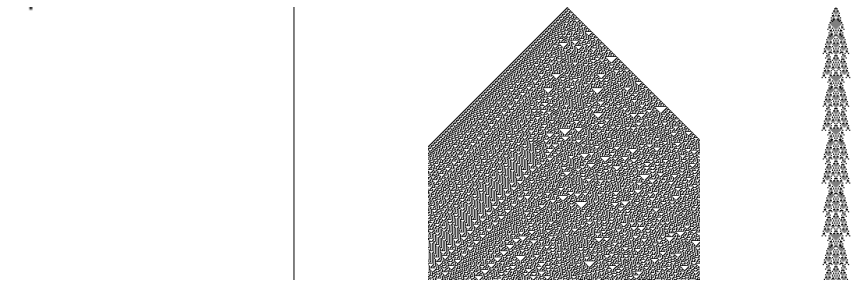


Figure 3: Illustration of different patterns generated by cellular automata.

Regarding applications, cellular automata are widely used when it involves self-reproduction, complex patterns and some randomness. For instance, they are used to model the growth of algae [Chen et al. 2002]; to detect eavesdroppers in wireless networks [Navid and Aghababa , Chap. 5]; to remove noise and to detect edges in digital images [Popovici and Popovici 2002]; to recognize patterns [Navid and Aghababa , Chap. 3]; and to model the evolution of certain vegetation [Ye et al. 2010], among others.

3. CABuilder – Cellular Automata Builder

Given the importance of cellular automata as a model of computation, a software tool called *Cellular Automata Builder* (CABuilder, for short) was developed in this work. The main goal of this tool is to provide a practical and simple way to generate graphical representations and pseudorandom sequences from cellular automata. In this first version of the software, the elementar and totalistic variants of such automata are available.

CABuilder was implemented according to the objected-oriented programming paradigm using the Python programming language (version 3.4). It makes use of the Python Image Library and of the dictionary data structure. The adoption of such programming language was motivated by its simple sintaxe and wide use. Since CABuilder is an open source software, such choices make easier future developments of this software.

To obtain graphical representations of cellular automata in CABuilder, the user must give four input parameters: (i) the type of cellular automaton, if it is elementar or totalistic; (ii) the number of sites of the automaton, which correspond to the width of the

final image; (iii) the rule that will determine how the automaton will evolve; and (iv) the number of iterations, which is the number of time steps the rule will be performed. This last parameter correspond to the height of the final image. In particular, if the automaton is a totalistic one, the user must provide an additional parameter that correspond to the amount of colors that will be used.

The output of CABuilder for such case is a graphical image from the evolution of a cellular automaton chosen by the user. This graphical representation is an image file according to the *Portable Network Graphics* (PNG) format. This file format is widely used and thanks to its support to lossless data compression, the colors of the automaton are reliable in the output file. The Figures 1b, 2b and 3, for instance, were produced with CABuilder.

Regarding pseudorandom sequences, the first work which considered such application of cellular automata was proposed by Wolfram [Wolfram 1986]. He describes some ways of mapping a vector of bits to an integer and then, from a given integer characterized by a finite vector of bits, to generate a new number [Gentle 2003]. Since this seminal work, several strategies to produce pseudorandom numerical sequences from cellular automata were proposed in the literature [Kang et al. 2008, Bardell 1990, Guan and Tan 2004]. In CABuilder, we follow the original approach to generate pseudorandom sequences from cellular automata proposed by Wolfram because it is the basis for most of the adaptations required by recent work.

To generate pseudorandom sequences with CABuilder, user must input analogous parameters as those to generate graphical images of cellular automata. Besides, user can choose if a separator must be used or not. If so, he must decide what separator to use, such as a blank space, a new line, a special character, etc. The adoption of a separator is particularly useful when the resulting sequences are used as input for other algorithms, for instance. The sequences produced are saved in a plain text file.

Besides those functionalities already mentioned, in CABuilder a user can generate a pseudorandom sequence from an existing graphical representation of a cellular automata and can edit a text file with a pseudorandom sequence by introducing separators or changing the numbers notation (from decimal to binary, for example). Both functionalities aim at allowing reuse from previous executions of this software.

The user interface of CABuilder where the previously mentioned functionalities can be accessed is shown in Figure 4. CABuilder executable and downloadable version, source files and its documentation are available in `https://goo.gl/lQdRfz`. Thanks to it license sharing, other researchers can introduce modifications and improvements in this software, such as batch image generation, new variants of cellular automata, among others.

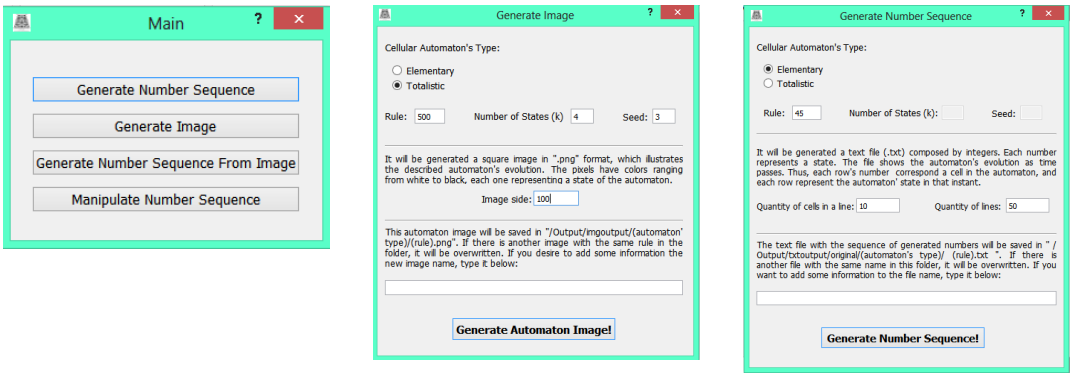
3.1. Preliminary Assessments of CABuilder

During the development phase some ad-hoc test procedures were adopted in order to ensure quality in the software version available to users. Many input and output tests were performed by the developer, by hand or in automated approach. They aimed at capturing all possible situations in user interaction with the tool.

Regarding correctness in the development of the computational model, graphical representations of the entire class of elementar cellular automata was produced, resulting in 256 image files. They were compared with canonical results available in the literature, such as those listed in Wolfram [Wolfram 2002]. The same strategy was adopted with

Figure 4: Screenshots of Graphics User Interface of CABuilder.

(a) Main Window **(b) Graphic Representation** **(c) Pseudorandom Sequences**



totalistic cellular automata with fixed $k = 3$, but given the high quantity of different automata in this category (3^7), just certain rules were considered, specially those of type-iv. No disagreements were found in the analysis performed.

Considering the ease of use, an executable version is available with a graphics user interface. Many instructions are presented along the user input, and error messages are displayed in alert windows, avoiding the execution with wrong parameters. Output files are available in pre-determined folders whose path is shown in the user interface.

Taking performance into account, it can be said that it is very reasonable. Most of results obtained during the development were in the order of seconds. It is a consequence of the quadratic order of growth, i.e., it takes $O(s \times t)$ time to build a cellular automaton of such variants, were s is the number of sites and t is the amount of time steps in evolution.

4. Related Work

In this section we are going to describe some related work found in the literature. Despite some existing library and software which address cellular automata, we are focused on existing work which enables creation of graphical images of cellular automata specially of the two variants previously discussed in Sec. 2. Besides, when possible, we will consider the generation of pseudorandom sequences.

Mathematica is a software mainly used for scientific, engineering, mathematical and computational purposes. It is based on symbolic mathematics and has its own library for cellular automata development [Gaylord and Nishidate 1996]. In this software, a celular automaton is created using the syntax of Mathematica and is instantiated with three parameters: the rule; a list that contains the initial state of the automaton; and the number of steps. The rule can be an integer in the case of an elementar automaton, but can also be other programming elements that enable the creation of totalistic celular automata and even other variants of this computational model. For a graphical representation, the resulting list of lists must be plotted and configuration regarding background color must also be provided. To generate pseudorandom sequences, instead of generating a graphical image, user must redirect the numbers from the resulting list to an output file. Programming in Mathematica is also needed for such task.

CAME&L is an workspace for cellular automata studying. It is composed by a C++ class library which is contains four main parts to develop a cellular automata: a

grid that implements visualization of sites; metrics which provide relationship among neighbor sites; datum that maintains cells states storage, exchange and some aspects of data visualization; and rules that fully describes computations and controls the iteration. CAME&L provides support for distributed cluster computations, but can also run on a personal computer. A graphics user interface enables visualization of results. Knowledge of C++ is required for implementing arbitrary cellular experiments [Naumov 2004]. No information regarding generation of pseudorandom sequences were found in this software documentation.

In both works, it is important to notice that there is a very extensive support to cellular automata. In the first work, the configuration is one of the highlights, while in the second work the parallelization is distinguished. However, both of them have some drawbacks regarding the demand for using programming languages to enable the creation of automata. In the first case, it is more critical because the syntax of the programming language is not so spread among the community as the one adopted in the second work.

When compared to the CABuilder, Mathematica and CAME&L require more effort in the creation of graphical representation of cellular automata because they require that a programming language must be learnt. CABuilder asks just the required information via a graphical user interface, but lacks very detailed user configuration in such input. For that reason, it can be said that CABuilder focuses on the simplicity to obtain the graphical representations of cellular automata. Other advantages of CABuilder is that it does not demand knowledge on programming nor programming languages; and that it is completely free and open source.

5. Final Remarks

In this work we introduced CABuilder, a software tool implemented in Python whose main objective is the generation of graphical representations and of pseudorandom sequences from cellular automata. It considers the elementar and totalistic variants of this computation model. CABuilder is a free and open source software that can be used by researchers in cellular automata; for teachers and students interested in this computational model; and also for developers who demand pseudorandom sequences in their applications

In future works, we aim at improving the graphics user interface of CABuilder. Besides, we would like to increase the number of variants of cellular automata; to allow different initialization configurations; and to create a framework to make it easier the addition of new kinds of cellular automata from third parties. All these improvements intent to cover a wider range of cellular automata to CABuilder users.

Acknowledgements

Authors acknowledge the financial support provided by Fundação de Amparo à Pesquisa do Estado do Amazonas (FAPEAM). The author Nicoli Pinheiro de Araújo is a scientific initiation fellow from PAIC $2014 - 2015$ from UEA/FAPEAM.

References

Bardell, P. H. (1990). Analysis of cellular automata used as pseudorandom pattern generators. In *International Test Conference*, pages 762–768.

Chen, Q., Mynett, A., and Minns, A. (2002). Application of cellular automata to modelling competitive growths of two underwater species chara aspera and potamogeton pectinatus in lake veluwe. *Ecological Modelling*, 147(3):253–265.

de Moraes, A. L. S. (2007). Um estudo sobre a aplicação de autômatos celulares na simulação de fenômenos ambientais e aspectos dinâmicos. Master's thesis, Universidade Católica de Pelotas, Pelotas, Rio Grande do Sul.

Fernández, M. (2009). *Models of Computation – An Introduction to Computability Theory*. Springer.

Gaylord, R. J. and Nishidate, K. (1996). *Modeling Nature: Cellular Automata Simulations with Mathematica*. Springer.

Gentle, J. E. (2003). *Random Number Generation and Monte Carlo Methods*. Springer, United States.

Guan, S.-U. and Tan, S. (2004). Pseudorandom number generation with self-programmable cellular automata. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(7):1095–1101.

Guedes, E. B. and Lula Jr., B. (2010). *Autômatos Finitos – com uma introdução aos Autômatos Finitos Quânticos*. Editora da Universidade Federal de Campina Grande, Campina Grande, Paraíba, Brasil.

Kang, B.-H., Lee, D.-H., , and Hong, C.-P. (2008). *Novel Algorithms and Techniques in Telecommunications, Automation and Industrial Electronics*, chapter Pseudorandom Number Generation Using Cellular Automata, pages 401–404. Springer.

Naumov, L. (2004). Came&l - Cellular Automata Modeling Environment & Library. *Lecture Notes in Computer Science*, 3305:735–744.

Navid, A. H. F. and Aghababa, A. B. *Emerging Applications of Cellular Automata*. In-Tech, United States.

Perrin, D. (1995). Les Debuts de la Theorie des Automates. *Technique et Science Informatiques*, 14:409–433.

Popovici, A. and Popovici, D. (2002). Cellular automata in image processing. In *Fifteenth International Symposium on Mathematical Theory of Networks and Systems*, volume 1.

Sarkar, P. (2000). A brief history of cellular automata. *ACM Computing Surveys*, 32(1):80–107.

Schiff, J. L. (2010). *Cellular Automata: A Discrete View of the World*. Wiley-Interscience, Campina Grande, Paraíba, Brasil, 1 edition.

Wolfram, S. (1984). Cellular automata as models of complexity. *Nature*, 311(5985):419–424.

Wolfram, S. (1986). Random sequence generation by cellular automata. *Advances In Applied Mathematics*, 7:123–169.

Wolfram, S. (2002). *A New Kind of Science*. Wolfram Media, Champaign, IL.

Ye, F., Chen, Q., and Li, R. (2010). Modelling the riparian vegetation evolution due to flow regulation of lijiang river by unstructured cellular automata. *Ecological informatics*, 5(2):108–114.