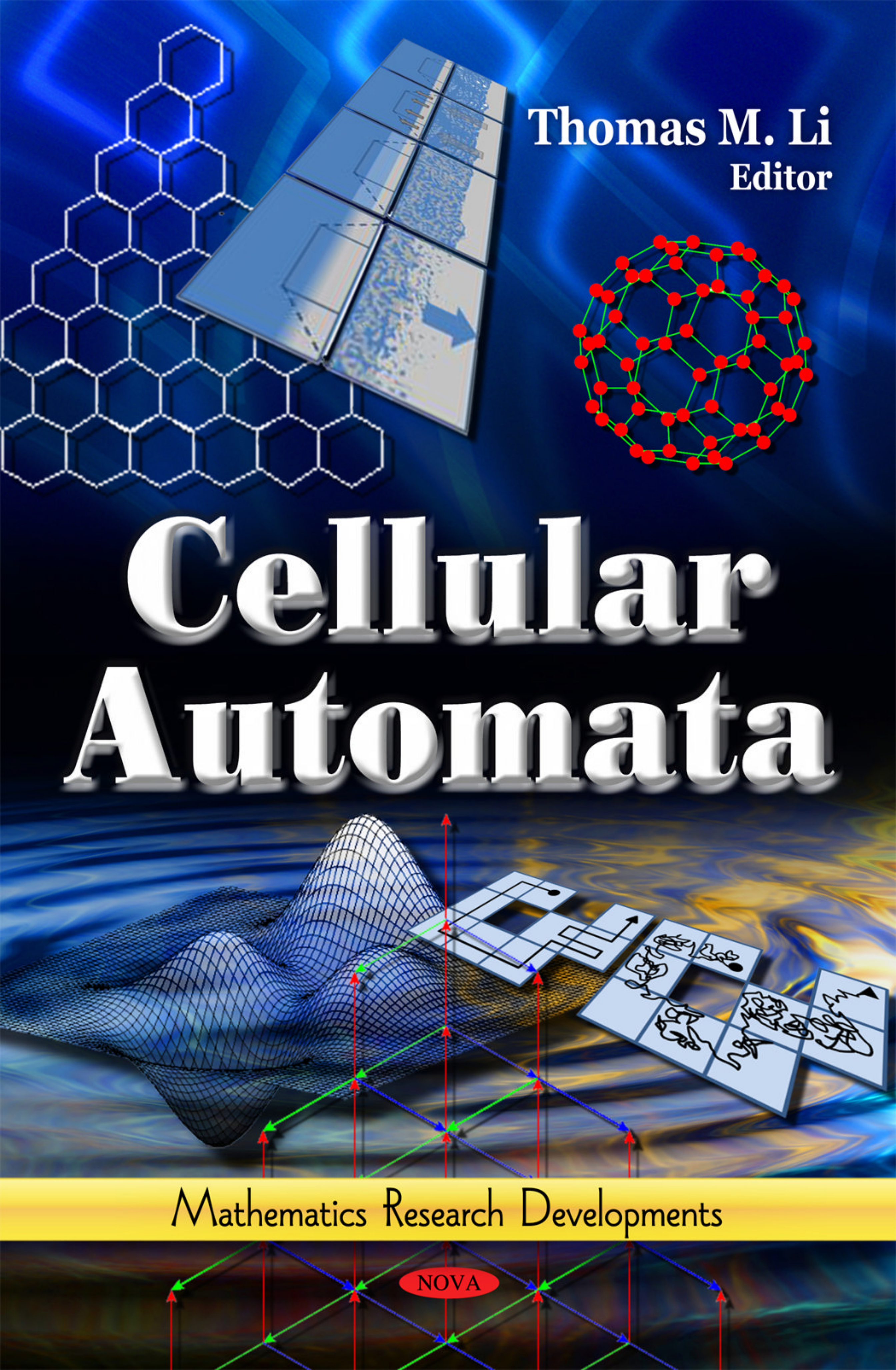


Thomas M. Li
Editor

Cellular Automata

Mathematics Research Developments

NOVA

MATHEMATICS RESEARCH DEVELOPMENTS

CELLULAR AUTOMATA

MATHEMATICS RESEARCH DEVELOPMENTS

Additional books in this series can be found on Nova's website
under the Series tab.

Additional E-books in this series can be found on Nova's website
under the E-book tab.

COMPUTER SCIENCE, TECHNOLOGY AND APPLICATIONS

Additional books in this series can be found on Nova's website
under the Series tab.

Additional E-books in this series can be found on Nova's website
under the E-book tab.

MATHEMATICS RESEARCH DEVELOPMENTS

CELLULAR AUTOMATA

THOMAS M. LI
EDITOR

Nova Science Publishers, Inc.
New York

For permission to use material from this book please contact us:
Telephone 631-231-7269; Fax 631-231-8175
Web Site: http://www.novapublishers.com

NOTICE TO THE READER

The Publisher has taken reasonable care in the preparation of this book, but makes no expressed or implied warranty of any kind and assumes no responsibility for any errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of information contained in this book. The Publisher shall not be liable for any special, consequential, or exemplary damages resulting, in whole or in part, from the readers' use of, or reliance upon, this material. Any parts of this book based on government reports are so indicated and copyright is claimed for those parts to the extent applicable to compilations of such works.

Independent verification should be sought for any data, advice or recommendations contained in this book. In addition, no responsibility is assumed by the publisher for any injury and/or damage to persons or property arising from any methods, products, instructions, ideas or otherwise contained in this publication.

This publication is designed to provide accurate and authoritative information with regard to the subject matter covered herein. It is sold with the clear understanding that the Publisher is not engaged in rendering legal or any other professional services. If legal or any other expert assistance is required, the services of a competent person should be sought. FROM A DECLARATION OF PARTICIPANTS JOINTLY ADOPTED BY A COMMITTEE OF THE AMERICAN BAR ASSOCIATION AND A COMMITTEE OF PUBLISHERS.

Additional color graphics may be available in the e-book version of this book.

LIBRARY OF CONGRESS CATALOGING-IN-PUBLICATION DATA

Cellular automata / [edited by] Thomas M. Li.
p. cm.
Includes index.
ISBN 978-1-62100-148-5 (eBook)
1. Cellular automata. I. Li, Thomas M.
QA267.5.C45C425 2010
511.3'5--dc22
2010031738

Published by Nova Science Publishers, Inc. † New York

CONTENTS

PREFACE

A cellular automaton is a discrete model studied in computability theory, mathematics, physics, complexity science, theoretical biology and microstructure modeling. It consists of a regular grid of cells, each in one of a finite number of states, such as "On" and "Off". The grid can be in any finite number of dimensions. For each cell, a set of cells called its neighborhood (usually including the cell itself) is defined relative to the specified cell. This book presents current research from across the globe in the study of cellular automata, including using cellular automata to solve optimization problems; modeling drug release science using cellular automata; using the cellular automata model to study the dispersion of aphids and ladybugs in a block of citric trees; and the reversibility of cellular automata.

Chapter 1 - One of the main problems in science extent that always are interested is optimization. The optimization problem is important subject that needs in many of science problems. “Optimization models” are a type of algorithm intended to provide the best possible solution to some problem facing an organization. Where the problem itself is so complex that finding the best possible solution could cost more than the benefit of doing so, the optimization models generally do not attempt to find the best possible solution, but instead seek to find extremely good solutions within reasonable cost and time parameters. This in fact is the more common situation.

A lot of different methods are proposed for solving optimization problem that each one has some advantages and disadvantages.

Regard to this subject that recently Cellular Automata was developed in investigation of complex systems, the approach in this study was getting the way for using cellular automata to solve the optimization problems. Due to its ability in simulating the local information while taking neighboring effects into account, the cellular automata technique is a powerful tool for optimization.

Cellular automata can be described in several ways. The description, which is perhaps most useful for physics, is to think of a CA as an entirely discrete version of a physical field. Space, time, field variables, and even the dynamical laws can be completely formulated in terms of operations on a finite set of symbols.

Cellular Automata (CAs) are one of the simplest models of highly parallel systems based on local rules. They were initially proposed as models of systems and processes made up of identical, simple, and locally interacting components. Researchers in this field used the simple models to study pattern formation and self-organization processes. It has been discovered that very complex pattern of behavior can be produced out of a set of a very simple rules. Recently it has been suggested that cellular automata and other simple programs may better model nature’s most essential mechanisms better than traditional mathematical equations.

One of the fundamental features of cellular automata, which make them highly useful computational tools for large systems, is their inherent parallelism. By assigning a simple processor to every so many cells of a large system of cells, one can increase the detail or the size of the system without increasing the time it takes to update the entire system. There does not seem to be a theoretical limitation or an overhead associated with splitting the problem into small pieces and distributing it. Thus, cellular automata simulations are very useful for massively parallel computers that are equipped with the proper hardware requirements.

In their modern implementation, cellular automata are viewed as simple mathematical idealizations of natural systems, and are used successfully to represent a variety of phenomena such as diffusion of gaseous systems, solidification and crystal growth in solids, and hydrodynamic flow and turbulence. In most of the previous applications, they are used to represent macroscopic behavior of a system, which are governed by partial differential equations of the continuum under consideration. This is generally accomplished using simple rules that represent the micro-mechanics of the medium [3].

In view of the above discussion, it is believed that CAs represent suitable tools for solving optimization problem.

Some useful efforts to adopt the cellular automata's concept, structure and dynamic for obtaining a suitable ability of optimization problem solving were performed, that their results compare with the other main optimization method such as neural network, and published in the relative journals.

Flexible and adaptable structure with complex physical system, reasonable calculation time, simplicity in implementing and ability to find good solutions near as comparable with global minimum are the most privileges of this method compare with former works.

In this chapter, the new and innovative procedures for extending the ability of optimization problem solving will review and describe.

Chapter 2 - Pharmaceutical science is a multidisciplinary field, in which fundamental and applied sciences unite to study all aspects related to drugs and pharmaceutical products. Amongst the specialities of this field, drug delivery focuses on altering the method of administration of therapeutic molecules to improve their effectiveness and safety. Various strategies exist to limit the toxicity or decrease the frequency of administration of an active ingredient. For example, polymeric biodegradable matrices are able to sustain and control the release of the drug. These systems are drug-reservoirs which are usually implanted subcutaneously or intramuscularly to discharge their load over a few weeks to a few months. The polymer implants may be fashioned in all kinds of shape: from films or larger blocks to micro- or nanosized particles.

The engineering of these formulations is complex and requires the combined expertise of polymer science, physical chemistry, and biology. In this context, cellular automata have emerged as interesting tools to model the various phenomena involved in the successful design and performance of these drug delivery systems. The following mini-review proposes a look at the different models available to depict the phenomena implicated in the release of drugs from biodegradable polymeric matrices.

Chronologically, the first aspect approached with the use of cellular automata was polymer erosion. The present work describes the various models proposed over the years, and emphasizes on the differences between them, from the probabilistic two-dimensional systems of the 1990's to the more comprehensive models proposed recently. In the second part of the manuscript, the authors focus on drug diffusion in the matrix, which is a phenomenon elegantly modeled by cellular automata. The various techniques used to describe molecular Brownian motion through confined space are also presented. Special attention is awarded to the role of the cellular automata to highlight the importance of these aspects on drug release, and to demonstrate that adequate modeling can achieve better understanding of these complex pharmaceutical systems.

Throughout this work, the authors wish to present a critical view of the objectives achieved through the use of cellular automata as model and bring to light the impending improvements which remain to be accomplished to consolidate their role in drug delivery science.

Chapter 3 - In this chapter the authors have adopted the Cellular Automata model (CA) to study the dispersion of the aphids and ladybugs in the block of citric trees. The main aim of this investigation has been to develop a simple and specific methodology to study Citrus Sudden Death (CSD). CSD is a disease that has affected sweet orange trees grafted on *Rangpur lime* in the state of São Paulo - Brazil. Some studies suggest that this disease has been caused by a virus and it is transmitted by insects known as *aphids* (vector). The *ladybug* was selected among the most known enemies of aphids in citrus in Brazil. In order to elaborate a predator-prey type of model to study the interaction between aphids (preys) and ladybugs (predators) in citriculture the authors have used a fuzzy rule-based system (FRBS). Qualitative information from specialists permit to propose rules that relate the state variables with their own variations. The states of the variables of the system (inputs) are the density of preys and the potentiality of predators and their variations are the outputs. To study the temporal evolution of the disease the authors have adopted a (CA) model. Therefore, the authors take into account the effect of he wind in the space covered by the aphid, since the wind is important for the flight of the aphid. Simulations were performed and compared between blocks with the presence of aphids and the absence of ladybugs, and the presence of both aphids and ladybugs. Numerical simulations allow us to foresee the behavior of the system, hence creating a spectrum of possibilities and proposing control techniques for different initial scenarios.

After this introduction, the chapter proceeds as follows: Section 1 develops brief reviews of Citrus Sudden Death, Cellular Automata and the concept of fuzzy set and fuzzy rule-based system. Section 2 details cellular automata model. Simulations results are in Section 3.

Chapter 4 - Cellular automata have been utilized for modeling and simulation of systems involving spatial arrangements. Examples include physical, social, urban and economic systems. The simulations are effected by means of local transition rules.

Cellular automata have also been employed as computation devices. For that purpose, genetic algorithms and simulated annealing were applied in order to determine suitable transition rules that would permit the cellular automaton to perform certain computational tasks. That approach constituted a form of optimization process.

However, in recent years optimization in a more general sense was presented in order to determine optimal configurations or arrangements of a cellular automaton. The latter may represent a physical or engineering system to be reformed or redesigned with a view to chosen objectives.

In the present approach, no specific transition rule is sought, but local transitions are guided by an evolutionary method suitably embedded into the automaton. Problems of spatial optimization have recently been treated by the author according to this methodology. In particular, location - allocation problems have been presented related to groundwater management, as well as land use management problems with or without allocation of resources.

The present communication proposes the application of an alternative evolutionary process based on mutating one by one all the components of the basic chromosome and selecting the best appearing result. This constitutes a local search to be combined with the special cell-based genetic algorithm described above. A general cellular – genetic framework is given for an algorithm suitable for spatial optimization problems. In an example problem, the superior performance of the proposed algorithm is shown.

Chapter 5 - Cellular Automata are often used as methodology to approach complex optimization problems in the Artificial Intelligence area, where some parts of the problem can be performed in a parallel way. In many cases, the complex nature of the problem can require high computational effort, so techniques that increase the performance are welcome. A possible solution that would allow us to accelerate the computation is to provide of a custom hardware processor designed for implementing several automaton running in parallel. To evaluate this alternative, the authors have designed and verified a reconfigurable processor that runs a stand-alone Cellular Automaton; hence many of these can work in parallel on the same chip. In order to obtain a valid prototype of the reconfigurable processor, the authors have considered the Conway’s Game of Life as example of a simple Cellular Automaton.

Chapter 6 - Form generation or morphogenesis has a crucial role in both artificial and natural development. This chapter presents results from simulations in which a genetic algorithm (GA) was used to evolve cellular automata (CA) in order to generate predefined 2D and 3D shapes. The 2D shapes initially considered were a square, a diamond, a triangle and a circle, whereas for the 3D case the shapes chosen were a cube and a sphere. The CA's rule was defined as a lookup table where the input was determined by the interaction neighborhood's cell state values, and the output established whether or not a cell was to be reproduced at the empty objective cell. Four different 2D interaction neighborhoods were considered: von Neumann, Moore, 2-Radial, and Margolus; a 3D Margolus neighborhood was used to generate the sphere and the cube. In all cases, the GA worked by evolving the chromosomes consisting of the CA rule table's output bits and a section of bits coding for the number of iterations that the model was to run. After the final chromosomes were obtained for all shapes, the CA model was allowed to run starting with a single cell in the middle of the lattice until the allowed number of iterations was reached and a shape was formed. The transition rules that formed some of these basic shapes and others were later combined with an Artificial (gene) Regulatory Network (ARN) to make up genomes that controlled the activation sequence of the CA's rules to generate predefined patterns. The ARN was also evolved by a GA in order to produce cell patterns through the selective activation and inhibition of genes. Morphogenetic gradients were used to provide cells with positional information that constrained cellular replication. After a genome was evolved, a single cell in the middle of the CA lattice was allowed to reproduce until a desired cell pattern consisting of the combination of basic forms was generated.

Chapter 7 - To study discrete dynamical systems of different types --- deterministic, statistical and quantum --- the authors develop various approaches.The authors introduce the concept of a system of discrete relations on an abstract simplicial complex and develop algorithms for analysis of compatibility and construction of canonical decompositions of such systems.

To illustrate these techniques the authors describe their application to some cellular automata. Much attention is paid to study symmetries of the systems. In the case of deterministic systems, they reveal some important relations between symmetries and dynamics. The authors demonstrate that moving soliton-like structures arise inevitably in deterministic dynamical system whose symmetry group splits the set of states into a finite number of group orbits. The authors develop algorithms and programs exploiting discrete symmetries to study microcanonical ensembles and search phase transitions in mesoscopic lattice models. The authors propose an approach to quantization of discrete systems based on introduction of gauge connection with values in unitary representations of finite groups --- the elements of the connection are interpreted as amplitudes of quantum transitions. The authors discuss properties of a quantum description of finite systems. In particular, they demonstrate that a finite quantum system can be embedded into a larger classical system. Computer algebra and computational group theory methods were useful tools in their study.

Chapter 8 - The authors establish a on-to-one correspondence between the configurations in the Wolfram cellular automaton, which is abbreviated to the WCA, and the paths in the de Bruijn quiver. Extending the correspondence to that between the associative algebra whose underlying vector space is generated by the configurations in the WCA and the path algebra of the de Bruijn quiver, the authors obtain the global transition of the associative algebra associated with the WCA. Thus the authors translate the problem concerning reversibility of the WCA into that concerning surjectivity of the endomorphism on the associative algebra. They then show that the induced problem concerning the endomorphism can be solved in terms of the adjacency matrix of the WCA, which is defined from that of the de Bruijn quiver through the one-to-one correspondence. Indeed, the authors give a necessary and sufficient condition for reversibility of the WCA. By virtue of the necessary and sufficient condition, the authors classify all 16 reversible rules in the ECA imposing periodic boundary conditions.

Chapter 9 - This paper deals with the emergence of computation in complex systems. The Turing universality (i.e. the ability to encompass the whole computation power of the class of Turing machines) of cellular automata which are the simplest representation of complex systems is considered.

The authors aim to construct an automatic system for the discovery of Turing-universal cellular automata. In this chapter, some steps towards this objective are presented, as is the search for self-localized patterns of non-resting states called gliders. An evolutionary method to search for gliders, based on a specific fitness functions taking into account the presence of periodic patterns and gliders, led to the discovery of a large number of gliders. Among the automata accepting gliders that were discovered, some would surprisingly generate glider guns for nearly every evolution of a random cell configuration. The first of them that was discovered is picked up as a potential candidate for a universal automaton.

Patterns that are able to stop unwanted streams of gliders are called eaters, they are searched for and used with gliders and glider guns to demonstrate the universality of an automaton.

Chapter 10 - This work presents a numerical implementation of a system that considers an encryption process of signals compressed by means of the Haar wavelet transform. An evaluation of the pseudorandom generator used in the encryption scheme, which is based on a cellular automaton, is carried out with some variants. In addition, the multifractal properties of the representative matrix of the generator are discussed.

Chapter 11 –The authors study factor subshifts, column factors and the canonical factor of cellular automata in a large setting, that is as endomorphisms of subshifts. Homogeneity of cellular automata makes them share some dynamical properties with their canonical factor; the authors review some of them.

In: Cellular Automata
Editor: Thomas M. Li, pp.1-37

ISBN: 978-1-61761-592-4

Chapter 1

CA UPGRADING FOR EXTENDING THE OPTIMIZATION PROBLEM SOLVING ABILITY

Amir Hosein Fadaei and Saeed Setayeshi
Faculty of Nuclear Eng. & Physics, Amirkabir University of Technology
(Tehran Polytechnique), Tehran, Iran

ABSTRACT

One of the main problems in science extent that always are interested is optimization. The optimization problem is important subject that needs in many of science problems. “Optimization models” are a type of algorithm intended to provide the best possible solution to some problem facing an organization. Where the problem itself is so complex that finding the best possible solution could cost more than the benefit of doing so, the optimization models generally do not attempt to find the best possible solution, but instead seek to find extremely good solutions within reasonable cost and time parameters. This in fact is the more common situation.

A lot of different methods are proposed for solving optimization problem that each one has some advantages and disadvantages.

Regard to this subject that recently Cellular Automata [1]was developed in investigation of complex systems [2], the approach in this study was getting the way for using cellular automata to solve the optimization problems. Due to its ability in simulating the local information while taking neighboring effects into account, the cellular automata technique is a powerful tool for optimization.

Cellular automata can be described in several ways. The description, which is perhaps most useful for physics, is to think of a CA as an entirely discrete version of a physical field. Space, time, field variables, and even the dynamical laws can be completely formulated in terms of operations on a finite set of symbols.

Cellular Automata (CAs) are one of the simplest models of highly parallel systems based on local rules. They were initially proposed as models of systems and processes made up of identical, simple, and locally interacting components. Researchers in this field used the simple models to study pattern formation and self-organization processes. It has been discovered that very complex pattern of behavior can be produced out of a set of a very simple rules. Recently it has been suggested that cellular automata and other

simple programs may better model nature’s most essential mechanisms better than traditional mathematical equations.

One of the fundamental features of cellular automata, which make them highly useful computational tools for large systems, is their inherent parallelism. By assigning a simple processor to every so many cells of a large system of cells, one can increase the detail or the size of the system without increasing the time it takes to update the entire system. There does not seem to be a theoretical limitation or an overhead associated with splitting the problem into small pieces and distributing it. Thus, cellular automata simulations are very useful for massively parallel computers that are equipped with the proper hardware requirements.

In their modern implementation, cellular automata are viewed as simple mathematical idealizations of natural systems, and are used successfully to represent a variety of phenomena such as diffusion of gaseous systems, solidification and crystal growth in solids, and hydrodynamic flow and turbulence. In most of the previous applications, they are used to represent macroscopic behavior of a system, which are governed by partial differential equations of the continuum under consideration. This is generally accomplished using simple rules that represent the micro-mechanics of the medium [3].

In view of the above discussion, it is believed that CAs represent suitable tools for solving optimization problem.

Some useful efforts to adopt the cellular automata's concept, structure and dynamic for obtaining a suitable ability of optimization problem solving were performed, that their results compare with the other main optimization method such as neural network, and published in the relative journals.

Flexible and adaptable structure with complex physical system, reasonable calculation time, simplicity in implementing and ability to find good solutions near as comparable with global minimum are the most privileges of this method compare with former works.

In this chapter, the new and innovative procedures for extending the ability of optimization problem solving will review and describe.

1. INTRODUCTION

Cellular automata (CA) form a general class of models of dynamical systems which are appealingly simple and yet capture a rich variety of behavior. This has made them a favorite tool for studying the generic behavior of and modeling complex dynamical systems. Historically CA are also intimately related to the development of concepts of computers and computation. This connection continues to be a theme often found in discussions of CA. Moreover, despite the wide differences between CA and conventional computer architectures, CA are convenient for computer simulations in general and parallel computer simulations in particular[2].

Thus CA have gained importance with the increasing use of simulations in the development of our understanding of complex systems and their behavior.

Space and time are both discretized, and the variables are often simplified to include only a few possible states at each site. Various cellular automata can be designed to model key properties of physical and biological systems.

John von Neumann was a central figure in the theory and development of automated computing machines. Originally, CA was introduced around 1950 (at the suggestion of

SlanislawUlam) in order to provide a simple model of self-reproducing automata. The successful (and profound) aim of this research was to show that certain essential features of biology could be captured in computational form. Much of von Neumann's work was completed and extended by Burks [4]. This line of CA research was followed through the 60's by related studies on construction, adaptation, and optimization as well as by investigations on the purely mathematical properties of CA.

A burst of CA activity occurred in the 70's with the introduction of John Conway's game of “life” [5]. Life was motivated as a simple model of an ecology containing 24 cells, which live and die according to a few simple rules. This most familiar example of a CA displays rich patterns of activity and is capable of supporting many intricate structures. In addition to its delightful behavior, the popularity of this model was driven in part by the increasing availability of computers, but the computers of the day fell well short of what was to come.

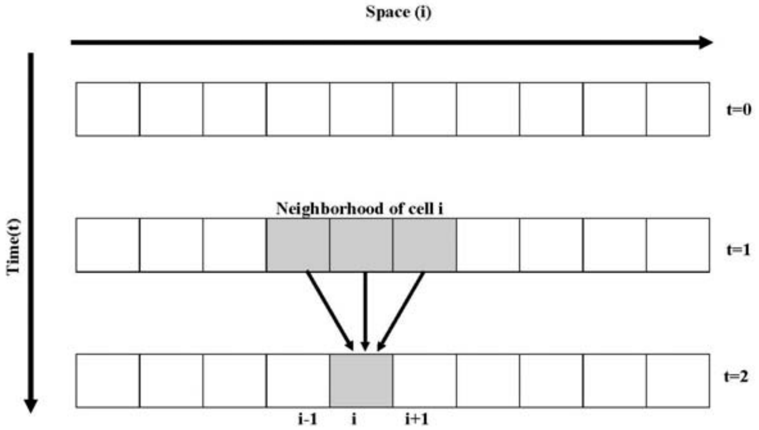


Figure 1. Space and time in a 1-dimensional CA

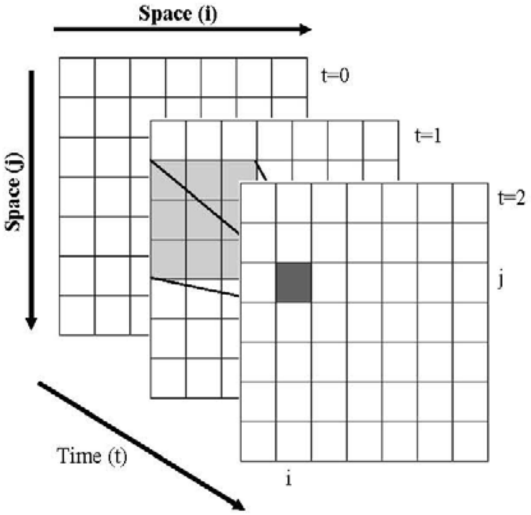


Figure 2. Space and time in a 2-dimensional CA

Cellular automata can be described in several ways. The description, which is perhaps most useful for physics, is to think of a CA as an entirely discrete version of a physical field. Space, time, field variables, and even the dynamical laws can be completely formulated in terms of operations on a finite set of symbols. The points (or cells) of the space consist of the vertices of a regular, finite-dimensional lattice, which may extend to infinity, though in practice, periodic boundary conditions are often assumed [6].

Time progresses in finite steps and is the same at all points in space. Each point has dynamical state variables, which range over a finite number of values. The time evolution of each variable is governed by a local, deterministic dynamical law (usually called a rule): the value of a cell at the next time step depends on the current state of a finite number of "nearby" cells called the neighborhood. Finally, the rule acts on all points simultaneously in parallel and is the same throughout space for all times [7]. Figures 1 and 2 show 1D & 2D examples of CA, respectively.

The concept of cellular automata begins from the concept of space and the locality of influence. We assume that the system we would like to represent is distributed in space, and that nearby regions of space have more to do with each other than regions far apart. The idea that regions nearby have greater influence upon each other is often associated with a limit (such as the speed of light) to how fast information about what is happening in one place can move to another place. Once we have a system spread out in space, we mark off the space into cells. We then use a set of variables to describe what is happening at a given instant of time in a particular cell.

$$\mathrm{s(i, j, k;t) = s(x_i, y_j, z_k ;t)} \tag{1}$$

where*i, j, k* are integers (*i, j, k*$\in$Z), and this notation is for a three-dimensional space (3-d). We can also describe automata in one or two dimensions (1-d or 2-d) or higher than three dimensions. The time dependence of the cell variables is given by an iterative rule:

$$\mathrm{s(i, j, k;t)} = \mathrm{R}\left(\left\{\mathrm{s}(\mathrm{i}' - i, \mathrm{j}' - j, \mathrm{k}' - k ; t - 1)\right\}\right)(\mathrm{i}', j', k' \in Z) \tag{2}$$

where the rule *R* is shown as a function of the values of all the variables at the previous time, at positions relative to that of the cell *s*(*i, j, k*;*t* - 1). The rule is assumed to be the same everywhere in the space, there is no space index on the rule. Differences between what is happening at different locations in the space are due only to the values of the variables, not the update rule. The rule is also homogeneous in time; i.e., the rule is the same at different times.

The locality of the rule shows up in the form of the rule. It is assumed to give the value of a particular cell variable at the next time only in terms of the values of cells in the vicinity of the cell at the previous time. The set of these cells is known as its neighborhood. For example, the rule might depend only on the values of twenty-seven cells in a cube centered on the location of the cell itself. The indices of these cells are obtained by independently incrementing or decrementing once, or leaving the same, each of the indices:

$$\mathrm{s(i, j, k;t)} = \mathrm{R}\left(s\,(i \pm 1, 0, j \pm 1, 0, k \pm 1, 0; t - 1)\right) \tag{3}$$

where the informal notation $i \pm 1,0$ is the set $\{i - 1, i, i + 1\}$. In this case there are a total of twenty-seven cells upon which the update rule $R(s)$ depends. The neighborhood could be smaller or larger than this example [2].

For describing more complicated system, more variables can be defined for each cell; more information is described in the next sections in details.

Cellular automata have high ability to simulate dynamic behavior of the complex systems. It is clear that the main case in applying Cellular Automata is the consistency with the system.It was mentioned that the CA can show the evolutionary procedure of a complex system; it means that there is a highly strong relation between the CA structure and the nature of the system. One important question can be arisen in this step:

How Can We Guide the System by CA?

If the answer of this question will find, we will gain the ability of guiding the system to our desire condition. In this chapter this question will be answered and the manner of guiding the system to optimal point described. Thus for obtaining the presented goal, several step should be passed.

In the second section the concept of complex system and its feature are presented to realize the nature of the systems. In the third section the optimization is presented conceptually, and the optimization procedure by CA is provided in section 4. Conclusion throughout the course of this study will be summarized in section 6.

2. COMPLEX SYSTEMS

The study of complex systems in a unified framework has become recognized in recent years as a new scientific discipline, the ultimate of interdisciplinary fields. It is strongly rooted in the advances that have been made in diverse fields ranging from physics to anthropology, from which it draws inspiration and to which it is relevant.

Many of the systems that surround us are complex. The goal of understanding their properties motivates much if not all of scientific inquiry. Despite the great complexity and variety of systems, universal laws and phenomena are essential to our inquiry and to our understanding. The idea that all matter is formed out of the same building blocks is one of the original concepts of science. The modern manifestation of this concept—atoms and their constituent particles—is essential to our recognition of the commonality among systems in science. The universality of constituents complements the universality of mechanical laws (classical or quantum) that govern their motion. In biology, the common molecular and cellular mechanisms of a large variety of organisms form the basis of our studies. However, even more universal than the constituents are the dynamic processes of variation and selection that in some manner cause organisms to evolve. Thus, all scientific endeavors are based, to a greater or lesser degree, on the existence of universality, which manifests itself in diverse ways.

In this context, the study of complex systems as a new endeavor strives to increase our ability to understand the universality that arises when systems are highly complex.

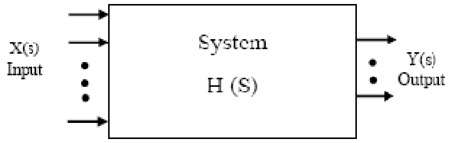


Figure 3. Schematic view of a sample system

A dictionary definition of the word “complex” is: “consisting of interconnected or interwoven parts.”Whythe nature of a complex system is inherently related to its parts? Simple systems are also formed out of parts. To explain the difference between simple and complex systems, the terms “interconnected” or “interwoven” are somehow essential. Qualitatively, to understand the behavior of a complex system we must understand not only the behavior of the parts but how they act together to form the behavior of the whole. It is because we cannot describe the whole without describing each part, and because each part must be described in relation to other parts, that complex systems are difficult to understand. This is relevant to another definition of “complex”: “not easy to understand or analyze.” These qualitative ideas about what a complex system is can be made more quantitative. Articulating them in a clear way is both essential and fruitful in pointing the way toward progress in understanding the universal properties of these systems.

For many years, professional specialization has led science to progressive isolation of individual disciplines. How is it possible that well-separated fields such as molecular biology and economics can suddenly become unified in a single discipline? How does the study of complex systems in general pertain to the detailed efforts devoted to the study of particular complex systems? In this regard one must be careful to acknowledge that there is always a dichotomy between universality and specificity.

A study of universal principles does not replace detailed description of particular complex systems. However, universal principles and tools guide and simplify our inquiries into the study of specifics. For the study of complex systems, universal simplifications are particularly important. Sometimes universal principles are intuitively appreciated without being explicitly stated. However, a careful articulation of such principles can enable us to approach particular systems with a systematic guidance that is often absent in the study of complex systems[2].

We can find many complex systems around us, such as our family in social systems or a human body from physiological point of view or other examples.

Now by thinking about the complex systems, what are the complex systems and what are their properties?

As before mentioned, simplest definition of the complex system is “hard to description”. The interconnections of the subsystems are very complicated in the complex systems; whereas each subsystem can’t define independent of the whole.

Several characteristics are defined for each complex system that can be used for classification or description of them and are listed as below:

- Elements (and their number)
- Interactions (and their strength)
- Formation/Operation (and their time scales)
- Diversity/Variability

- Environment (and its demands)
- Activity (ies) (and it's [their] objective[s])

The properties of complex systems can be quantified using analyzing their characteristics. The common calculation tools use of numerical description of the system for evaluating it; therefore, the system should be described numerically to identify itsbehavior. Transfer function is a Mathematical relation that is describingthe behavior of the system; and will be presented in the next in details.

In system identification procedure, each dynamic system can be described using transfer function. Transfer function is a relation that presents the nature of the system and can be used as a representation of the system. In the dynamic analysis, each system has an input and an output, the relation between the input and output is described by transfer function. If we summarize a sample system, which is shown in Figure 1, the X(s) is the set of input variables and Y(s) is an output variables, the transfer function can be defined as follow:

$$H(s) = \frac{Y(s)}{X(s)} \tag{4}$$

It is clear that transfer function has all the characteristics of the system in its nature. Finding the transfer function for dynamic system is the main part of identification procedures.

For optimizing the system, we should know the nature of the system; therefore we should find the transfer function. With obtaining the transfer function guiding the system to desired state will be possible, that the details of this procedure will described in the next sections.

3. OPTIMIZATION

Optimization is the act of obtaining the best result under given circumstances. In design, construction, and maintenance of any engineering system, engineers have to take many technological and managerial decisions at several stages. The ultimate goal of all such decisions is either to minimize the effort required or to maximize the desired benefit. Since the effort required or the benefit desired in any practical situation can be expressed as a function of certain decision variables, *optimization* can be defined as the process of finding the conditions that give the maximum or minimum value of a function.

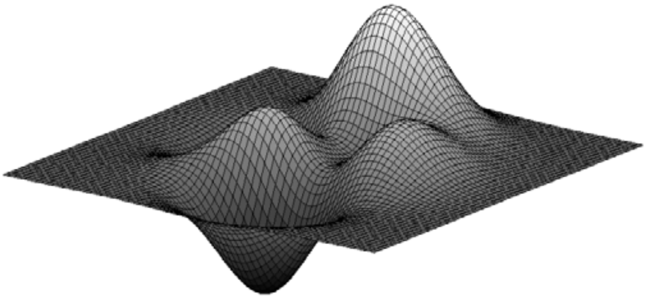


Figure 4. The sample surface for describing the nature of optimization

Optimization can be taken to mean minimization since the maximum of a function can be found by seeking the minimum of the negative of the same function. There is no single method available for solving all optimization problems efficiently. Hence a number of optimization methods have been developed for solving different types of optimization problems.

The nature of optimization problem can be described by considering a sample surface, that the maximum or minimum place of that is requested. Check the Figure 4, and find a good answer for the following questions:

- Which place is good for starting?
- How could moved for finding the best path?
- What is the benefit of passed path as experience for continuing?
- What is the best searching procedure?
- Where is the desired place?
- and ….

Now with summation of the whole themes that are coming from your intrinsic answers, the concept and required cased for understanding and describing the optimization procedure will be clear.

The optimum seeking methods are also known as *mathematical programming techniques* and are generally studied as a part of operations research[8]. *Operations research* is a branch of mathematics concerned with the application of scientific methods and techniques to decision making problems and with establishing the best or optimal solutions.

Mathematical programming techniques are useful in finding the minimum of a function of several variables under a prescribed set of constraints. Stochastic process techniques can be used to analyze problems described by a set of random variables having known probability distributions. Statistical methods enable one to analyze the experimental data and build empirical models toobtain the most accurate representation of the physical situation. This book deals with the theory and application of mathematical programming techniques suitable for the solution of engineering design problems.

3.1. History

The existence of optimization methods can be traced to the days of Newton, Lagrange, and Cauchy. The development of differential calculus methods of optimization was possible because of the contributions of Newton and Leibnitz to calculus. The foundations of calculus of variations, which deals with the minimization of functionals, were laid by Bernoulli, Euler, Lagrange, and Weirstrass. The method of optimization for constrained problems, which involves the addition of unknown multipliers, became known by the name of its inventor, Lagrange. Cauchy made the first application of the steepest descent method to solve unconstrained minimization problems. Despite these early contributions, very little progress was made until the middle of the twentieth century, when high-speed digital computers made implementation of the optimization procedures possible and stimulated further research on new methods. Spectacular advances followed, producing a massive literature on optimization techniques. This advancement also resulted in the emergence of several well-defined new areas in optimization theory.

It is interesting to note that the major developments in the area of numerical methods of unconstrained optimization have been made in the United Kingdom only in the 1960s. The development of the simplex method by Dantzig in 1947 for linear programming problems and the annunciation of the principle of optimality in 1957 by Bellman for dynamic programming problems paved the way for development of the methods of constrained optimization. Work by Kuhn and Tucker in 1951 on the necessary and sufficiency conditions for the optimal solution of programming problems laid the foundations for a great deal of later research in nonlinear programming. The contributions of Zoutendijk and Rosen to nonlinear programming during the early 1960s have been very significant. Although no single technique has been found to be universally applicable for nonlinear programming problems, work of Carroll and Fiacco and McCormick allowed many difficult problems to be solved by using the well-known techniques of unconstrained optimization. Geometric programming was developed in the 1960s by Duffin, Zener, and Peterson. Gomory did pioneering work in integer programming, one of the most exciting and rapidly developing areas of optimization. The reason for this is that most real-world applications fall under this category of problems. Dantzig and Charnes and Cooper developed stochastic programming techniques and solved problems by assuming design parameters to be independent and normally distributed[8].

The desire to optimize more than one objective or goal while satisfying thephysical limitations led to the development of multiobjective programmingmethods. Goal programming is a well-known technique for solving specific types of multiobjective optimization problems. The goal programming wasoriginally proposed for linear problems by Charnes and Cooper in 1961. Thefoundations of game theory were laid by von Neumann in 1928 and since thenthe technique has been applied to solve several mathematical economics andmilitary problems. Only during the last few years has game theory been appliedto solve engineering design problems.

Simulated annealing, genetic algorithms, and neural network methods representa new class of mathematical programming techniques that have comeinto prominence during the last decade. Simulated annealing is analogous tothe physical process of annealing of solids. The genetic algorithms are searchtechniques based on the mechanics of natural selection and natural genetics.Neural network methods are based on solving the problem using the efficientcomputing power of the network of interconnected "neuron" processors.

3.2. Objective Function

The conventional design procedures aim at finding an acceptable or adequate design which merely satisfies the functional and other requirements of the problem. In general, there will be more than one acceptable design, and thepurpose of optimization is to choose the best one of the many acceptable designs available. Thus a criterion has to be chosen for comparing the different alternative acceptable designs and for selecting the best one. The criterion, with respect to which the design is optimized, when expressed as a function of the design variables, is known as the *objective function.* The choice of objective function is governed by the nature of problem. Defining the suitable objective function is the main part of optimization problem solving; specially in complicated system optimizing.

An optimization problem involving multiple objective functions is known as a *multiobjective programming problem.* With multiple objectives there arises a possibility of conflict, and one simple way to handle the problem is to construct an overall objective function as a linear combination of the conflicting multiple objective functions. Thus if $g_1(x)$ and $g_2(x)$ denote two objective functions, construct a new (overall) objective function for optimization as

$$g(x) = a_1 g_1(x) + a_2 g_2(x) \tag{5}$$

where a_1 and a_2 are constants whose values indicate the relative importance of one objective function relative to the other.

Needless to say that the converge procedure of multiobjective optimization problem that uses linear combination is very dependent on the nature of the problem and also determination of weight for each object. In the different system, several variables are selected as the dependent variable in optimization procedure. It is clear that these parameters have different behavior, cost, values and weight; in decision making procedure, these parameters should be compared and finally the suitable values of them be selected based on defined desired situation. Comparing the different variables for reaching the desired situation isn't possible with attentiononly in their values, in the other word, the nature of this variable and their effects on the system situation should be take in to account in the comparing procedure.

3.3. System Optimization

Based on mentioned cases, it is clear that each system consists of several independent variables and usually one dependent variable. The independent variables define the basic characteristics of the system, and the dependent variable is used as indicator for describing the system situation. If an assumed system is described with the following equation:

$$f\big(X(t)\big) = H(x_1(t), x_2(t), \dots, x_n(t)) \tag{6}$$

$$X(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ \cdot \\ \cdot \\ \cdot \\ x_n(t) \end{pmatrix} \tag{7}$$

f(x(t)) is the dependent variable of the system that can be used for describing its situation, X(t) is the set of independent variables and the H(.) function is the relation that describe the effect of independent variables in the system situation. Needless to say that, the f(x(t)) function is the same as system transfer function and can be presented the complexity and inner relation of the system. Complex systems,regard to situation of the system that is expressed by independent variables, show different behavior. If the f variable is assumed one dimensional, Figure 5 can be used for describing the conceptual situation of the system based on different independent variables condition.

Figure 5 is schematically shown the dependent variable behavior versus changing the set of independent variables. Each physical system can be described by this definition, one indicator and several independent variables for presenting the condition of the system. This is very interesting that each system can be defined based on desired indicator while the subjected indicator can be selected among several dependent parameters based on desirable object, i.e., there are several different way to describe the system and selecting one of those ways dependent on required task of the system in analyzing procedure.

If we suppose a sample electrical circuit (shown in Figure 2) as a complex system, it will be clear that the elements are the individual parts of the system and cannot be analyzed separately. This system such as each complex system has several independent variables and a dependent variable to define and recognize. The set of x variables are defined as independent variables of the system and can be composed of:

- The circuit structure
- The type of sub-elements
- The number of each type of sub-elements
- The state of inter-connected between sub-elements
- and …

Based on intended task for the circuit, one dependent parameter can be defined, such as voltage or current, and called as f(x).It can be seen from Figure 6 that there are many different configuration for subjected circuit; 2 sample configurations are shown in Figure 6. Each configuration has its proprietary characteristic and is shown with X_i ,i=1,…n. X_i represents the set of independent variables for defining the situation of the system in i^{th} state.

With defining the suitable task for this circuit, we can obtain the best set of independent variables for reaching to the desired goal. Suppose that we want to have the high voltage as an output of this system, so voltage will be the dependent variable of the system and the sub-elements should be arranged for producing the high voltage in the output. Therefore different configuration of the system (circuit) should be analyzed for obtaining the best for desired task.

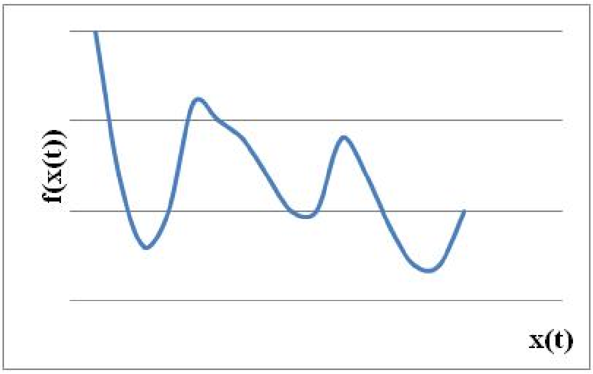


Figure 5. The behavior of the system versus different set of independent variables

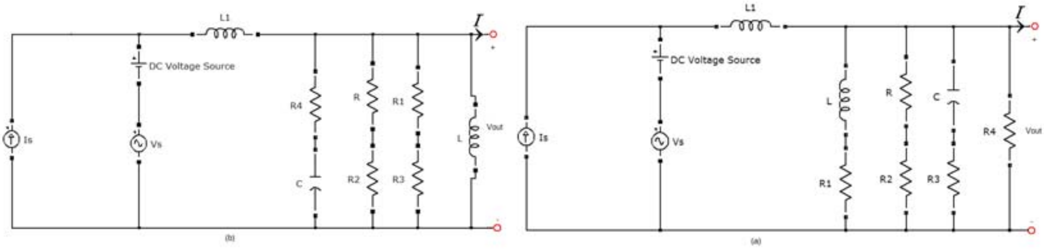


Figure 6. sample electrical circuit

In the system optimization problem, reaching to one desired situation of the system that analytically can be described by one or several parameters of the system is goaled. Recognitionand realization of the system perform based on following some main parameters that are defined as indicator. Therefore finding the suitable parameter that can be shown and described the behavior of the system is very important; needless to say that this parameter will be assumed as dependent variable of the system.

Now, we return to define the system parameters. In each system one desirable task is defined as its main work. Certainly, improving the quality and design of the system to perform its task as better as before would be high interested. If the independent variables of the system present the situation of the system that can be changed for obtaining the better revenue, thus, finding the best set of independent variables for guiding the dependent variable to desired value is called *system optimization procedure*.

4. OPTIMIZATION BY CA

It was mentioned that in designing procedure of different system the optimization is one the main cases that should be considered to obtain the best situation. In this section the new innovative method to solve the optimization problem using the cellular automata as a base tool is introduced.

Cellular automata is a calculation and simulation tool that can model the complex behavior of different system. Certainly,asserting that all of the systems and phenomena can be explained by mathematical relation isn't wrong. Therefore each system can be presented by a complex mathematical equation. Regard to the high ability of CA for simulating of complex behavior, each complex system can be modeled by CA. Needless to say that the main case in this issue is to define the suitable structure for consisting the system and CA.

There are two main concepts in cellular automata that are neighboring and transition rule. Also, in the optimization problem, there are two different procedures, which are decision making and changing procedure of independent variables.

In this chapter we try to define a new method to extending the optimization problem solving ability to CA using all of the CA's ability but step to step because of its complexity. Two different methods are introduced and described in the following, which are [9]:

1. Optimization by CA+SA
2. Optimization by CA

The main different between these two methods is in the circumstance of the system following. In the other word, independent variable is needed as indicator for recognizing the situation of the system. The method of obtaining the cost variable (independent variable) is one of the causes of different in two methods. The procedure of decision making in optimization procedure is one of the other cases that lead to different. In the first method the independent variable is obtained globally and the whole system is modeled and its equations solved tofind the output, whereas, in the second method all the system and variables are calculated locally, these cases will described in the next sections in more details.

These two methods are explained and finally used to solve the important complicated optimization problem to proof the validation of them.

4.1. Optimization by CA+SA

This section introduces a design methodology in the context of finding new and innovative design principles by means of optimization techniques. In this method Cellular Automata (CA) and Simulated Annealing (SA) were combined and used for solving the optimization problem. This method contains two principles that are neighboring concept from CA and accepting each displacement basis on decreasing of objective function and Boltzman distribution from SA that plays role of transition rule. Simulated Annealing Procedure will explain in the next section for best describing the method.

4.1.1. Simulated annealing

Undoubtedly the most developed of the stochastic optimization methods is SA [10, 11]. Casually, it appears to be very simple method to implement; however, the reality is that to obtain both a robust, i.e. consistently locate the family of near-optimum decisions, and efficient, i.e. minimizes number of histories that must be examined to locate the family of near-optimum decisions, some thought must be given. SA is based upon the analogy of a solid slowly cooling to its lowest energy state, i.e. annealing.

Given a minimization objective, let *E* denote the value of the objective function for the currently accepted decision variables, and *E* denote the value of the objective function after some perturbations are made to the currently accepted decision variables. Whether the perturbed decision variables are accepted, implying they now become the currently accepted decision variables, is determined by the following rules.

$$\begin{aligned} &if \quad E^* \le E \quad or \quad if \quad E^* > E \quad and \quad random[0,1] < e^{-(E^*-E)/KT} \\ &\qquad\qquad then \qquad E = E^* \\ &\qquad\qquad otherwise \quad E = E \end{aligned} \tag{8}$$

So if the perturbed decision variables produce a lower objective function value, then they become the currently accepted decision variables. However, if the perturbed decision variable produces a higher objective function value, they only become the currently accepted decision variables if a random number between zero and one exceeds the expression noted in Eq. (8); otherwise, the currently accepted decision variables are unaltered.

The conditional acceptance of inferior solutions allows the search algorithm to escape local minimums, and as we shall see when utilizing penalty functions, traverse the infeasible decision space. The parameter *T* plays the analog of material temperature in annealing. At high temperatures, many inferior solutions are accepted, allowing the search space to be extensively transverse in search of the vicinity of the global minimum. As temperature decreases, the probability of acceptance of inferior solutions decreases, since if the cooling schedule is done appropriately one should now be within the vicinity of the family of global optimum solutions. What the initial temperature should be and how fast cooling should occur determine the robustness and efficiency of the SA implementation. Fortunately, both of these attributes can be determined based upon the specific behavior of the optimization problem that is being solved as the search progresses [12, 13].

Another item that must be addressed is at what temperature cooling should be ended and the optimization search terminated. This can be triggered by a combination of lower temperature limit, maximum number of histories, and lack of improvement in the objective function value.

4.1.2. Procedure

In this section, as before mentioned, a new method for solving optimization problem has been introduced [14]. Proposed method contains two concepts that are Cellular Automata and Simulated Annealing. By combining of these concepts high ability for solving optimization problem will be got. To solve the optimization problem define and calculate the objective function are needed. In Eqs9 to 11 a typical objective function is introduced, needless to say that reach to optimum value of objective function is approached in optimization procedure.

$$E(X_1,...,X_g) = H(R_1(X_1),...,R_g(X_g)) \tag{9}$$

$$X_g = f_g(P(\{i,j,k\};t)) \tag{10}$$

$$P(\{i,j,k\};t) = V(S(\{i,j,k\};t)) \tag{11}$$

where:

E is an objective function value,

X_g is an optimization variable,

g is a number of optimization variable,

R_g is a function that follow X_g variable for optimization,

p is a cell value at t,

f_g is a function that follow X_g variable based on local information (cells values),

S is state of cell,

V is a function that describes the dynamic relation in the system and between cells,

$\{i,j,k\}$ is a set of cell,

H is an objective function based on linear combination of $\{R_l(X_l), l = 1,...g\}$

In this method three variables were defined for each cell that are cell state (S), cell value (P) and cell properties (L(u), u=1,…,v that v is the number of cell properties that are used in dynamic calculation of system). It is clear that the dynamic behavior of under consideration system is described by V function. P variable is the output value for each cell that is explaining cell condition resulted from studying the behavior of the system. The objective variable could be obtained from cell values by special relation that is named as f. In this procedure, V function describes the inner relation of the system between independent variables to create the dependent variable. P variable can be summarized as an independent variable of the system for each cell. By analyzing the P variables, system indicator, which is universally defined by E, can be calculated.

The main part of the optimization procedure is to define a suitable method for changing the independent variables whereas the system passing the short way until reaching to the optimum value. In this proposed method, a new evolution procedure based on S.A. and CA' neighboring is introduced that is imparting from local characteristics of the system to evolve.

Cell state is the independent variable that should be changed for optimizing the system. Changing procedure of the cell state to reach the optimum value is described in Eq12 to 15.

$$S(\{i,j,k\};t)=Q(S(\{i,j,k\};t-1)) \tag{12}$$

$$Q(S(\{i,j,k\};t-1))=\begin{cases} S(\{i',j',k'\};t-1) & if \quad ((\Delta E\le 0)\ OR\ ((\Delta E>0) AND (Random[0,1]<e^{\frac{-\Delta E}{KT(t-1)}}))) \\ S(\{i,j,k\};t-1) & otherwise \end{cases} \tag{13}$$

$$\Delta E = E\Big|_{S(\{i',j',k'\};t-1)} - E\Big|_{S(\{i,j,k\};t-1)} \tag{14}$$

$$T(t)=\alpha\, T(t-1) \tag{15}$$

where:

$\{i',j',k'\}$ is a set of cells when one of selected cell state is replacing with one of its neighbors that is selected randomly,

$T(t)$ is a temperature,

K is Boltzman constant, and

α is a constant that is named Annealing coefficient.

According to the mentioned equations, Q function determines the evolution procedure of cell state. This function analyzes the cell state replacement influence on objective function and determines the next value of cell state based on it.

Note that in this method several initial conditions should be defined that are:

$$S_0=S(\{i,j,k\};0)\ ;\ T(0)\ ;\ \alpha$$

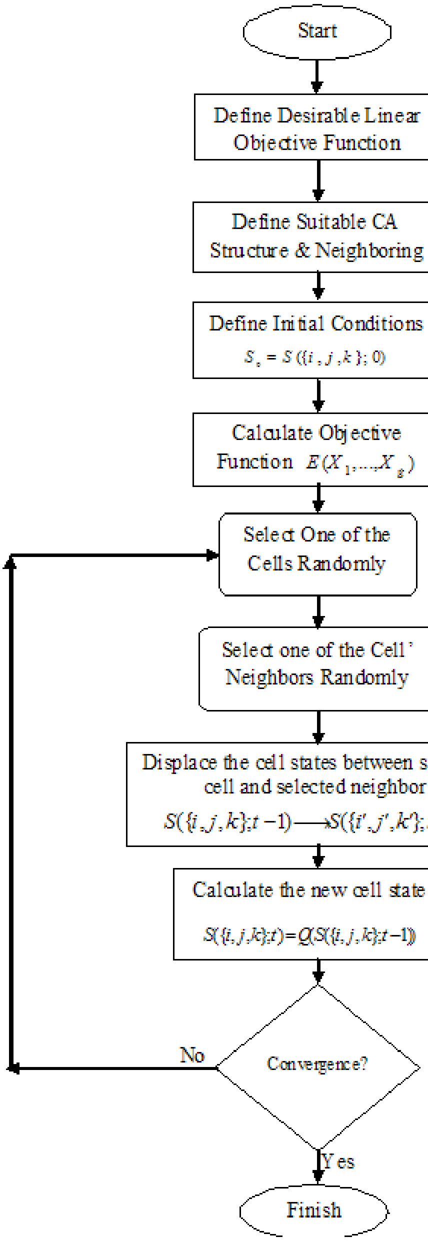

Start
Define Desirable Linear Objective Function
Define Suitable CA Structure & Neighboring
Define Initial Conditions
$S_c = S(\{i,j,k\};0)$
Calculate Objective Function $E(X_1,\ldots,X_g)$
Select One of the Cells Randomly
Select one of the Cell' Neighbors Randomly
Displace the cell states between selected cell and selected neighbor
$S(\{i,j,k\};t-1) \longrightarrow S(\{i',j',k'\};t-1)$
Calculate the new cell state
$S(\{i,j,k\};t)=Q(S(\{i,j,k\};t-1))$
Convergence?
No
Yes
Finish

Figure 7. Flowchart of CASA optimization method

Cellular Automata depicts physical structure of problem and neighboring of each element. Therefore the adaptation of CA and problem Structure is necessary. The role of Simulated Annealing is to escape the result from local minimum and guide it to global minimum as near as possible (Eq13). The procedures of this method are as follow and shown in the Figure 7 schematically.

1. Define an linear objective function that involves all goals with their suitable weights (Eq 9)

2. Define a suitable CA structure and neighboring of each cell for modeling the problem
3. Distribute all of elements in the CA randomly using a pre-define set of elements ($S_0 = S(\{i, j, k\}; 0)$)
4. Evaluate a value of objective function
5. Generate the sequence of cell number randomly, it means that if CA has N cells should produce sequence of N number that are 1,2,…,N and distribute randomly
6. Repeat (7) and (8) until all cells update (N iterations)
7. Select one of cell number from generated sequence basis on iteration number and displace it with one of its neighbor that selected randomly
8. Evaluate a value of objective function and run simulated annealing for judging about acceptability of displacement, if displacement is acceptable then the new statement of CA unchanged even if not acceptable then statement of CA change to last statement (Eq 13)
9. Repeat (5) to (8) until converge the CA, i.e., when objective function receive to local optimum and the statement of CA be unchanged through calculation.

One of the main advantages of this method is to have short computation time.

In this study proposed procedure is applied to solve the fuel management optimization problem in VVER-1000 reactor. Procedure and results of this study has been taken in next sections.

4.1.3. A Sample problem solving

Proposed method was used for solving fuel management optimization problem in VVER-1000 Russian reactor [15, 16]. Since the fuel management problem contains a huge amount of calculation for finding the best configuration for fuel assemblies in reactor core this method has been introduced for reducing the volume of calculation. In this study reducing of power peaking factor and increasing the cycle length inside the reactor core of Bushehr NPP is considered as the objective function. The result is the optimum configuration, which is in agreement with the pattern proposed by the designer.

In fuel management optimization, minimization of power peaking factor & maximization of initial excess reactivity are considerable objects. It means that the fuel assemblies should be arranged in the core in such a manner that the neutron flux is flattened as high possible and also has a maximum initial excess reactivity. Flatness of neutron flux causes to increase reactor safety margin and well distribute of fuel burnup; increasing the excess reactivity can increase the life time of reactor that is important case in economic. Therefore in this study, minimum value of power peaking factor and maximum value of initial excess reactivity in VVER-1000 reactor core are approached.

Objective function is as follow (Eqs16 to 18), needless to say that this objective function can be changed and the other parameters exceeded on:

$$E(F_R, K_{ef}) = A/2R_1(F_R) + B/2R_2(K_{ef}) \tag{16}$$

$$R_1(F_R) = (F_R - 1)^2 \tag{17}$$

$$R_2(K_{ef}) = \left(1 - K_{ef}\right)^2 \tag{18}$$

Where, A, B are constants, and F_R and K_{ef} are obtained by performing the dynamic calculation in the core. It is clear that F_R and K_{ef} are as X_1 and X_2 in the optimization procedure and can be obtained by solving dynamic relation of the system. Diffusion equation is validated for describing the dynamic relation equation in the nuclear reactor core; thereforediffusion equationand power distribution can be considered as V functionand P variable, respectively.

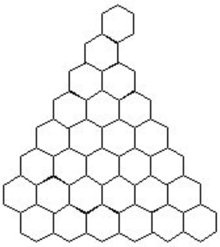


Figure 8. Physical structure of CA

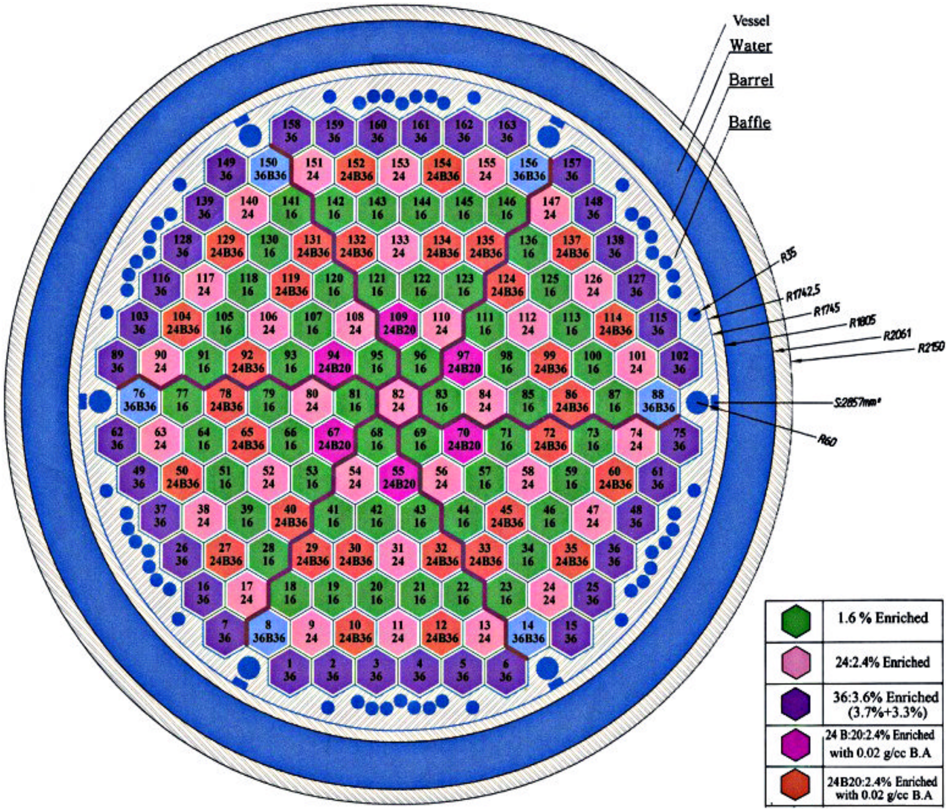


Figure 9. Core shape of VVER-1000 reactor proposed by designer

At the first step mapping of the fuel management problem onto the CA & SA is depicted. For this, at first defining the suitable structure of CA that adapt to the physical structure of VVER-1000 reactor core is needed. VVER-1000 reactor core has hexagonal shape (Figure 8) thus hexagonal cell for making CA structure is defined. Physical structure of CA is as shown in Figure 8. Since the VVER-100 reactor core has 1/6 symmetry (Figure 9). The second step is to define the neighboring of each cell in CA. In this problem each cell is neighbored with 6 cells that are around it (Figure 10), needless to say, that the cells in periphery of CA structure have less than 6 neighbors.

In this study, the positions of 7 fuel assemblies in the periphery of the core and the central one are fixed as boundary conditions (Figure 11); therefore should determine the best positions for remaining 20 assemblies.

After defining the CA structure and neighboring is needed to make the set of elements that should distribute in core as fuel assemblies. Therefore enrichment of each fuel assembly and burnable absorber that may be in one fuel assembly should be prepared as a set of elements by user. In this study the set of elements that the designer of VVER-1000 reactor proposed was used. In the next step elements that prepared are distributed over the cells randomly, i.e. each fuel assembly in the set of elements is located in one cell of CA, here we are 20 fuel assemblies and 20 empty cells. Note that in this study the kind of fuel assemblies that are mentioned in table 1 are selected as cell states. Therefore 6 number of fuel assemblies' kind are made the collection of cells states that are presented in table 1.

Since there are 20 cells in CA should be generated the sequence of 1, 2, ..., 20 randomly, that updating of each cell is performed basis on this sequence.

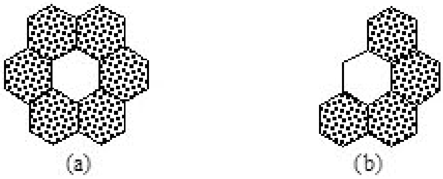


Figure 10. Cell in CA structure and its neighbors. (a) Cell with 6 neighbors, (b) cell with 4 neighbors in periphery of CA structure

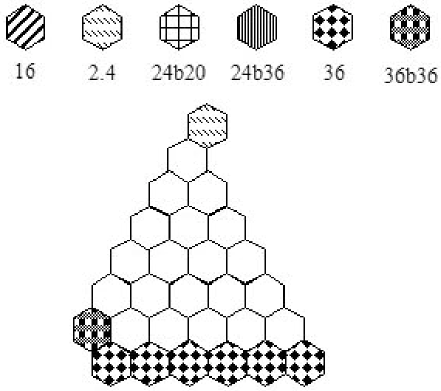


Figure 11. Structure of CA with initial condition

Table 1. Type of fuel assemblies in VVER-1000 reactor core (Cell States collection for CA)

No.	Fuel Assembly's Abbreviation	Description
P=1	16	Fuel Assemblies of 1.6% enriched fuel rods
P=2	24	Fuel Assemblies of 2.4% enriched fuel rods
P=3	36	Fuel Assemblies of 3.7% & 3.3% enriched fuel rods
P=4	24B20	Fuel Assemblies of 2.4% enriched fuel rods & burnable absorber rods with a density of 20 kg/m^3
P=5	24B36	Fuel Assemblies of 2.4% enriched fuel rods & burnable absorber rods with a density of 36 kg/m^3
P=6	36B36	Fuel Assemblies of 3.7% & 3.3% enriched fuel rods & burnable absorber rods with a density of 36 kg/m^3

At this step one of the cells is selected, basis on sequence made, and displaced fuel assembly in this cell with fuel assembly that is one of its neighbors, which is selected randomly. Then neutronic calculation is performed by new pattern and objective function is calculated. The new and old value of objective function is given to simulated procedure for making a decision about acceptability of new pattern, if new pattern is acceptable then remains unchanged, if not acceptable new pattern is displaced with old pattern. This procedure is introduced as transition rule for CA.

This procedure continued until converged, i.e. when the value of objective function remains constant.

The method outlined in this paper could be applied to the other objective functions, which are not considered here.

We have applied the CA model accompanied by SA to find the optimum-loading pattern for first cycle of Bushehr NPP and compare it with the pattern proposed by the designer [4].

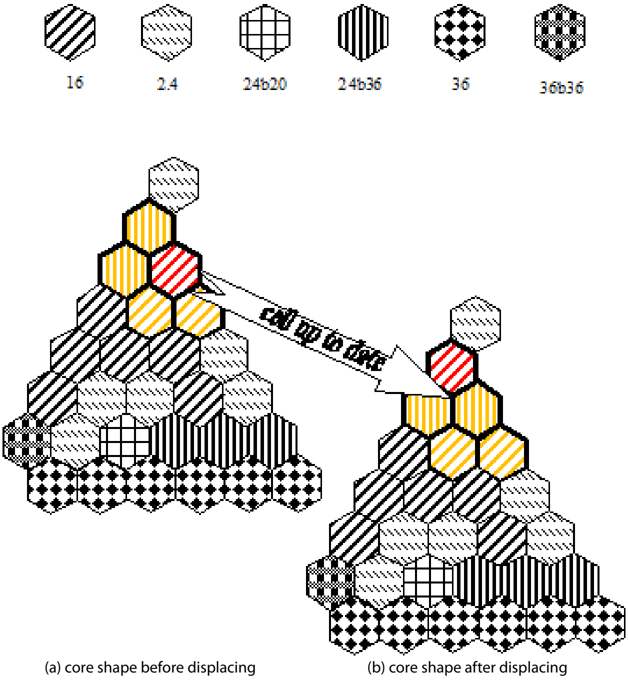
Proposed method has program that use FORTRAN software as base media. This program has several subroutines that are:

- CA structure
- Simulated Annealing
- Neutronic calculation

CA structure subroutine contains structure of Cellular Automata and state of each cell. Besides this section perform the random displacement for each cell with its neighbors.

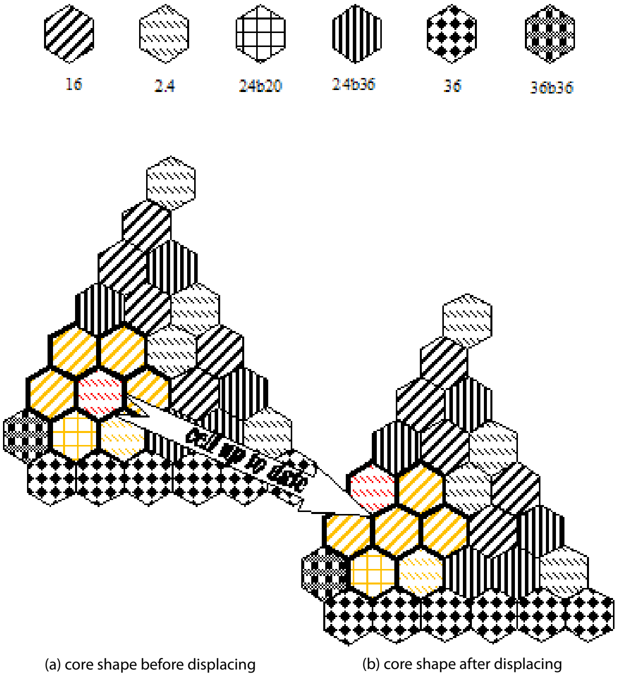
Simulated Annealing subroutine performs the algorithm, which escape the network from local minimum and guide to global minimum. In fact, in this subroutine evaluation of the acceptability of each displacement in CA's state is performed.

In Figures 12 to 15 evolution of the calculation by CA and SA program from initial state to final state, when it is converged to an acceptable solution, is shown. As it is expected the solution found by CA & SA is global minimum of energy function.



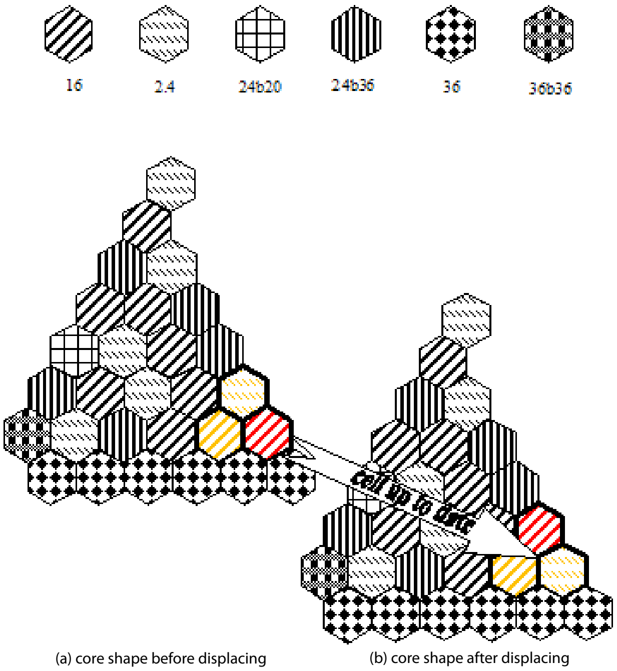
(a) core shape before displacing (b) core shape after displacing

Figure 12. CA & SA process for first loading of Bushehr NPP – first iteration (Temperature=100°K, E=78.2172, $F_R = 1.58$ & $K_{ef} = 1.1603$)



(a) core shape before displacing (b) core shape after displacing

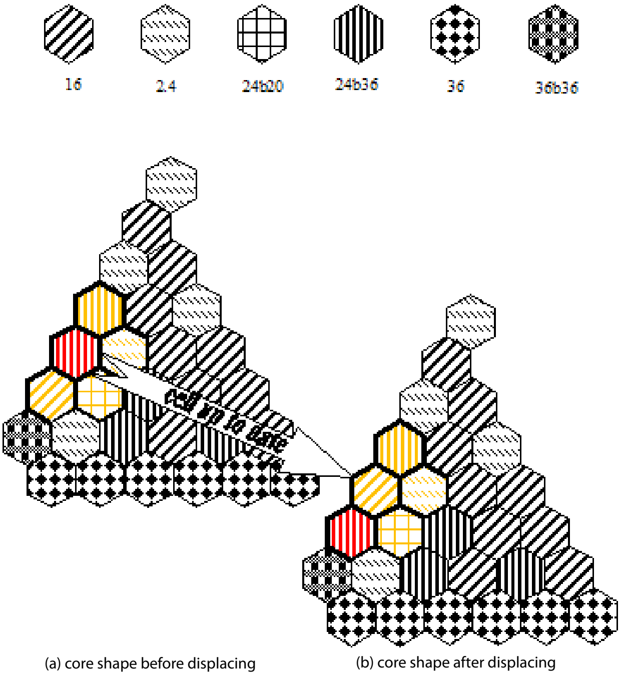
Figure 13. CA & SA process for first loading of Bushehr NPP – 17th iteration (Temperature=52.040°K, E=33.4868, $F_R = 1.39$ & $K_{ef} = 1.1574$)



(a) core shape before displacing

(b) core shape after displacing

Figure 14. CA & SA process for first loading of Bushehr NPP – 58th iteration (Temperature=9.7602°K, E=22.5429, $F_R = 1.33$ & $K_{ef} = 1.1569$)



(a) core shape before displacing

(b) core shape after displacing

Figure 15. CA & SA process for first loading of Bushehr NPP – last iteration (Temperature=4.3140°K, E=15.9319, $F_R = 1.30$ & $K_{ef} = 1.1625$)

Start SA's subroutine with initial point at a small temperature ($T_{in} = 100°K$) and reduce it gradually in each step (annealing coefficient = 0.96). In this way reaching to the global minimum is assured in acceptable time. Figures 12 to 15 show the process of reaching to the global minimum. Note that in section (a) of these figures red cell is selected cell for updating and yellow cells are its neighbors, in section (b) accepted replacement of cell and selected neighbor is shown.

In the table 2, we have calculated F_R and k_{ef} for the two core configurations which has been discussed. Evolution of energy and objective parameters in proposed method for fuel management optimization of VVER-1000 reactor core had been shown in Figures 16 & 17, respectively.

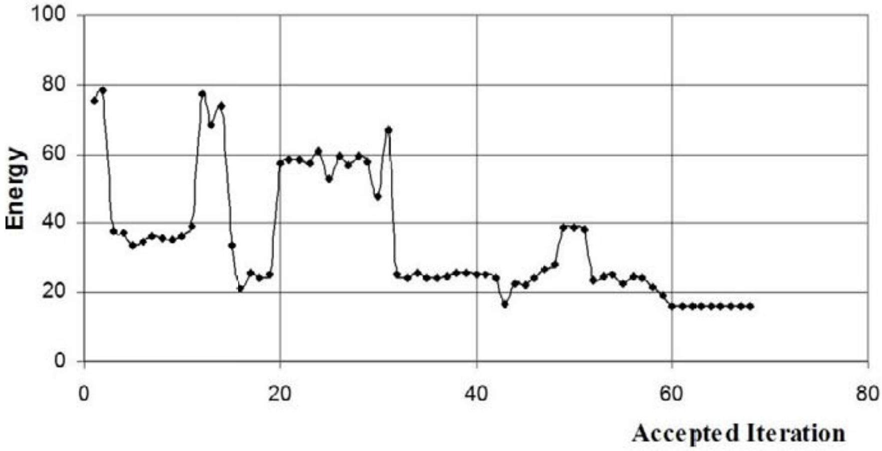


Figure 16. Evolution of energy in the CA & SA procedure for first loading of Bushehr NPP

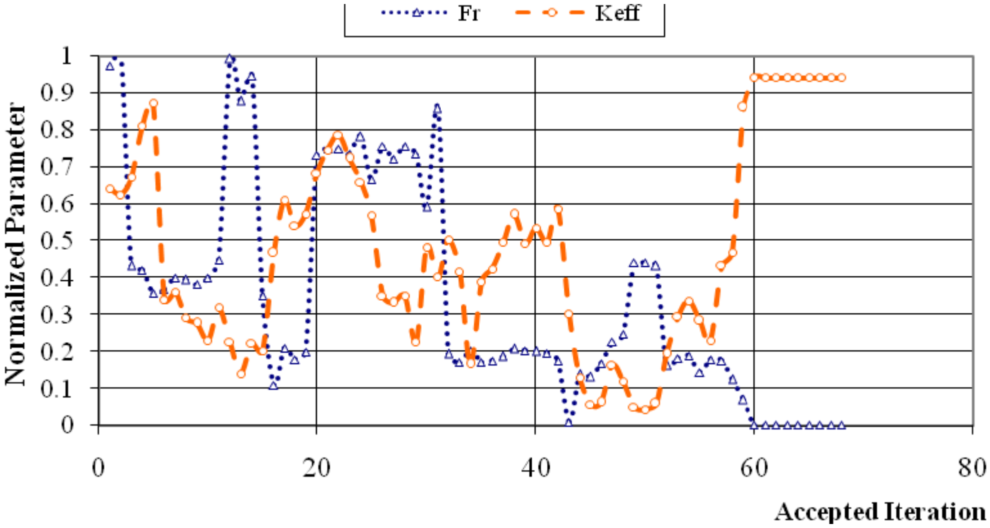


Figure 17. Evolution of objective parameter in the CA & SA procedure for first loading of Bushehr NPP

If the objective parameters of the end configuration, that were given by proposed method, Figure 15 (F_R = 1.30, K_{ef} = 1.1625), are compared to the parameters of the configuration proposed by VVER Russian Designer (Figure 18), which are F_R = 1.39 & K_{ef} = 1.1577, we will conclude that the solution found by the proposed method is closer to optimum.

As shown in table 2 calculated result by combination of CA & SA for loading optimization as compared to the configuration, which proposed by VVER-1000, have lower power peaking factor and greater initial excess reactivity; therefore this method has good reliability for finding the optimum core configuration.

Needless to say, that by improving the objective function and exceeding the parameters that have important role in fuel management and loading pattern the final result improved.

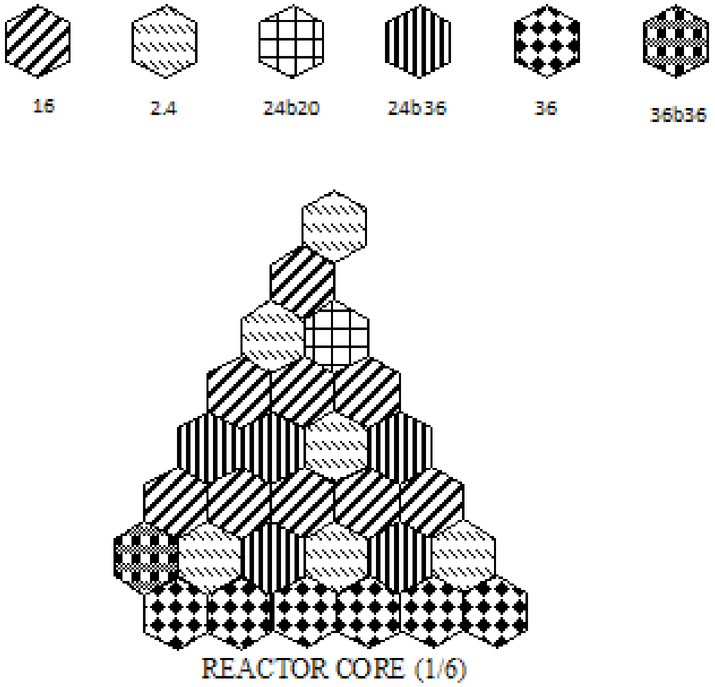


Figure 18. Configuration that is proposed by VVER for first loading of Bushehr NPP – (F_R = 1.39 & K_{ef} = 1.1577)

Table 2. Comparison of results

Method	F_R	K_{ef}	No. of Iterations
CASA	1.30	1.1625	77
VVER-1000	1.39	1.1577	-

4.2. Optimization by CA

In the previous section, the new method that is used from CA accompanying with simulated annealing for solving optimization problem was introduced. In that method the total ability of CA for simulating the behavior and finally guiding the system to the optimal weren't used. In this section the other method is introduced that is used from the CA without any other concept [17]. The main characteristics of the CA, neighboring and locality, are used in this method and the system be evolved to an optimal point based on local information.

4.2.1. Procedure

In this new method, physical structure in a problem and the neighboring of each element should be simulated by cellular automata; therefore adaptation of the CA structure with considered system in a problem is necessary.

To solve an optimization problem, one needs to define and calculate an objective function. In the optimization procedure, the optimum value of the objective function is approached. Since updating of each cell in CA depends on the state of its neighboring cells, the main component of this method is in fact the definition of the objective function using neighboring cells parameters. Therefore, each variable that is to be optimized should be defined using neighboring cells parameters which are based on local information. This means that a transition rule function, that describes the local relationship in CA, should be used for defining the objective function.In the proposed method, procedure of cells updating should be performed in a manner that cause to guide the system to desirable situation. Each cell is updated based on defined transition rule in the CA, if we want to guide the system to desired situation we should consider a suitable transition rule that changes the cell state based on local information and evolves the system to the optimum state.

Based on mentioned cases in the previous sections, each system has a special structure with its relations. In this method, the structure and the relation of the different parts of the system should be obtained whereas lead to the best desired state, therefore defined CA with its transition rule should be contained all of the characteristics of the subjected system.

In the proposed method, two parameters are considered for each cell which are state and kind. The cell state is describes the value of parameter that will be used for defining the objective function. Note that this parameter is obtained by calculating the dynamic behavior of system. The cell kind describes the value of parameter that has been changed for obtaining the optimum solution. In other words this is an independent variable parameter that changes the value of the objective function. In fact the final solution of optimization is a set of cell kinds for the system that are led to the optimum value in objective function.

Briefly, each system has a treatment that is described with a relation. This relation has some independent variables as an input and a dependent variable as an output. Input and output variables are defined as cell kinds and cell states, respectively. Needless to say that, the goal of optimization procedure is to find the best value of independent variables (cell kinds) that are led to optimum value of the objective function. Finally the set of cell kinds (cell kind for all of cells) is approached for optimization problem. In the proposed method for kind of each cell, m different values are possible. Needless to say that, if there are q cells in the system, m^q different conditions (set of cell kinds) are possible in optimization procedure;

finding the best possible condition that is led to optimum value of objective function is approached.

In Eqs.19 and 20 a typical objective function based on local information and the procedure of changing cell kinds are introduced.

$$E(X_1,...,X_g) = \min\{H_p(R_{1,p}(s),...,R_{g,p}(s)); p = 1,...,m\}$$
$$= H_y(R_{1,y}(s),...,R_{g,y}(s)) \tag{19}$$

$$R_{g,p}(s(\mathrm{i, j, k;t})) = f_g(s(\{i',j',k'\};t-1), n_{l,p}(\{i,j,k\};t-1)) \tag{20}$$

$$n(\{i,j,k\};t) = n_{l,y}(\{i,j,k\};t-1) \tag{21}$$

$$n(\{i,j,k\};0) = n_0 \tag{22}$$

In these equations, E is an objective function, X_g is an optimization variable, g is the number of optimization variable, $R_{g,p}(s)$ is a function that follow X_g variable based on local information by neighboring states for the p 'th cell kind for (i,j,k) cell , f_g is a rule function that follow X_g variable based on local information by neighboring states, $\{i',j',k'\}$ is a set of cell and its neighbors, n({i,j,k};t) is a set of cells kind at time t, $n_{l,y}$({i,j,k};t) is a set of cells kind at time t that the kind of l 'th cell was changed to y, l is a number of cell that is marked with (i,j,k), H_p is an objective function for p 'th kind of cell, m is the number of possible kinds of cell, y is the best cell kind for (i,j,k) cell, and s is the state of the cell.

The schematic flowchart of this method is introduced in Figure 19 and its procedure is as follow:

1. Define a linear objective function that involves all goals with their suitable weights
2. Define a suitable CA structure and neighboring of each cell for modeling the problem
3. Define a set of cells kind as an initial condition for CA (n_0), randomly
4. Generate the sequence of cell numbers randomly, i.e., if CA has N cells, we should produce the sequence of N numbers 1,2,…,N, and distribute them randomly
5. Repeat the forthcoming steps (6) to (10) until all cells are updated (N iterations)
6. Select one of the cell numbers from the generated sequence
7. Repeat steps (8) to (9) for all of cell kinds (m iterations)
8. Evaluate the value of rule functions ($f_1,...,f_g$)
9. Calculate H_p from $R_{g,p}(s)$ based on the important degree and other relationships depending on the optimization variables
10. Determine the best cell kind (y) by finding the minimum value of H_p,p=1,…,m (The cell kind that is caused to minimum value of objective function is selected as a final solution in this iteration)

11. Repeat steps (4) to (10) until convergence is achieved, i.e., until the objective function arrives at a local optimum and the set of cell kinds that are led to optimum value of objective function.

The mentioned procedure can be described and understood clearly in a sample problem solving procedure, sothis proposed procedure for optimization is applied for solving the fuel management optimization problem in VVER-1000 reactor, too.

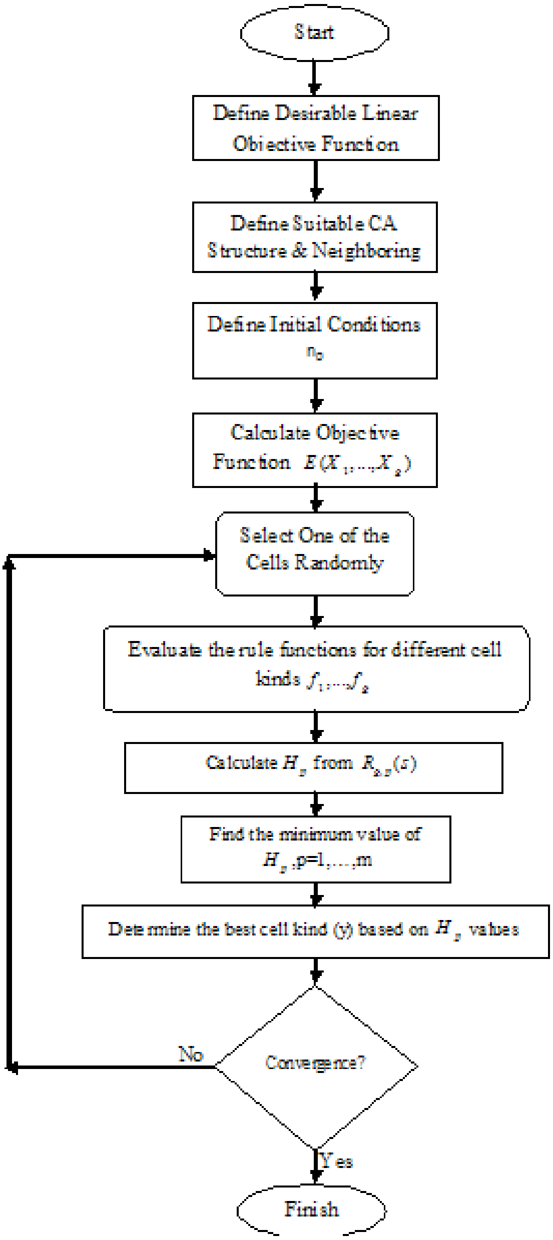

Start
Define Desirable Linear Objective Function
Define Suitable CA Structure & Neighboring
Define Initial Conditions n_0
Calculate Objective Function $E(X_1,...,X_z)$
Select One of the Cells Randomly
Evaluate the rule functions for different cell kinds $f_1,...,f_z$
Calculate H_p from $R_{z,p}(s)$
Find the minimum value of H_p ,p=1,…,m
Determine the best cell kind (y) based on H_p values
No
Convergence?
Yes
Finish

Figure 19. Flowchart of CA optimization method

4.2.2. A Sample problem solving

In this section, like to previous solved problem, the fuel management optimization procedure with power peaking factor because of its dependence on safety considerations the initial excess reactivity that can impress the cycle length of the reactor core are considered. The objective function and transition rule in this study are defined as follow:

$$E(F_R, K_{ef}) = \min\{H_p(R_{1,p}(s), R_{2,p}(s)); p = 1, \dots, m\} \tag{23}$$

where $R_{1,p}(s)$ and $R_{2,p}(s)$ are the functions that follow F_R and K_{ef} parameters, respectively . The value of m depends on the defined structure and cell kinds, and that will be described in the next section. Due to the nature of CA, updating of the cells should be performed based on local information, therefore f_1 and f_2 were defined for relating $R_{1,p}(s)$ and $R_{2,p}(s)$ functions to local information. f_1 and f_2 were found by performing data analysis procedure in the reactor core model. Depending on problem structure, different lattices are defined; the parameter z represents the number of cells in the selected lattice (the number of neighbors+1).

$$\begin{aligned} R_{1,p}(s(\mathrm{i,j;t})) &= f_1(s(\{i',j'\};t-1),p) \\ &= \sum_{k=1}^{z}\sum_{w=k+1}^{z}\left|\{(s(\{i',j'\}_k;t-1)|_{n_{l,p}(\{i,j,k\};t-1)}) - (s(\{i',j'\}_w;t-1)|_{n_{l,p}(\{i,j,k\};t-1)})\}\right| \end{aligned} \tag{24}$$

Here, the set $\{i',j'\}_k$ shows the k 'th cell and its neighbors. In this study the state of each cell was defined as cell power in the reactor core. As mentioned before, the function $R_{1,p}(s)$ should follow the power peaking factor, decreasing this parameter means that the fuel assemblies should be arranged in the core in such a manner that the neutron flux is flattened as much as possible, therefore this function was defined based on the flatness of the neutron flux or power (Eq. 24).

$$\begin{aligned} R_{2,p}(s(\mathrm{i,j;t})) &= f_2(s(\{i',j'\};t-1),p) \\ &= \frac{\sum_{k=1}^{z}(s(\{i',j'\}_k;t-1)|_{n_{l,p}(\{i,j,k\};t-1)})}{z*(s(\{i,j\};t-1)|_{n_{l,p}(\{i,j,k\};t-1)})} \end{aligned} \tag{25}$$

Based on data analysis, the relationship that is shown in Eq. 25 follows the K_{ef} parameter; this means that by decreasing this function K_{ef} parameter is increased.

Processing of the data for decision making in optimization procedure is performed by the function H_p. This function is defined based on the significance of each parameter in the optimization procedure. In some of optimization procedure, the objective variables have different nature in the behavior, values and cost; therefore making the suitable objective function is very difficult. In the procedure of defining the comparison weight for optimization variables, different methods can be used; the main point in this issue is to obtain the similar base for all parameters to gain the comparison ability. The procedure of making suitable objective function in optimization is very complicated and spacious and there isn't appropriate opportunity for describing in more details in this chapter.Anyway for this problem the function was innovatively defined and described as follow:

$$H_p(R_{1,p}(s), R_{2,p}(s)) = A * \exp(R_{1,p}(s)^*) + B * \exp(R_{2,p}(s)^*) \tag{26}$$

The parameters A and B are selected based on the important degree of the related parameter in the optimization. $R_{1,p}(s)^*$ and $R_{2,p}(s)^*$ are normalized values of $R_{1,p}(s)$ and $R_{2,p}(s)$, respectively, so that their values remain between 0 and 1:

$$R_{g,p}(s)^* = \left(\frac{1}{\max\{R_{g,p}(s), p = 1,...,m\} - \min\{R_{g,p}(s), p = 1,...,m\}}\right) * (R_{g,p}(s) - \min\{R_{g,p}(s), p = 1,...,m\}) \tag{27}$$

After calculating the function $H_p(R_{1,p}(s), R_{2,p}(s))$ for p=1,…,m, the decision for selecting the best kind of cell in the optimization procedure is made by finding the minimum value of H_p (Eq. 23).

In this method, the power is defined as the state of each cell, therefore one needs to simulate the reactor core and perform neutronic calculations. For performing neutronic core calculations WIMS-D4 and CITATION codes should linked, for this purpose a FORTRAN software was employed [16].

Such as previous section the CA structure select based on the system structure and since the VVER-1000 reactor core has 1/6 symmetry (Figure 9), therefore we only need to model 1/6 of the core.

In this method the set of fuel assemblies (cell kinds) that is led to optimum value of objective function is approached. It is clear that, the kind of cell describes an enrichment of fuel that is placed in that cell. In this study, the kind of 7 cells in the periphery of the core and the central one are fixed (Figure 11) as initial condition; therefore the best kind for remaining 20 cells should be determined.

After defining the CA structure and the neighborhood of the cells, one needs to define the initial conditions for the cells. This can be performed randomly. Therefore kind of each cell (enrichment of fuel) selected from set of different enrichment of fuel assemblies that is shown in table 1. In this study 5 th and 6 th kinds of fuel assemblies only for defining boundary condition and the other for optimization procedure are used, therefore m=4 in this problem.

In the next step, the sequence of 1,2,...,20 should be generated randomly (updating of each cell is performed basis on this sequence). At this step one cell is selected, basis on sequence made, then using Eqs21 to 25 and performing neutronic calculation, objective function is calculated and the cell is updated (best kind of cells, suitable enrichment, for this cell is determined).

This procedure continued until converged, i.e. when the value of objective function remains constant.The method outlined in this paper could be applied to the other objective functions, which are not considered here.

We have applied the CA model to find the optimum-loading pattern for first cycle of Bushehr NPP and compare it with the pattern proposed by the designer (VVER-1000).

First fuel assembly's group constants prepared by WIMS D-4 code which are functions of temperature, burnup and power. The results are used in input file of CITATION code [15, 16].

Proposed method has program that use FORTRAN software as base media. This program has two subroutines that are:

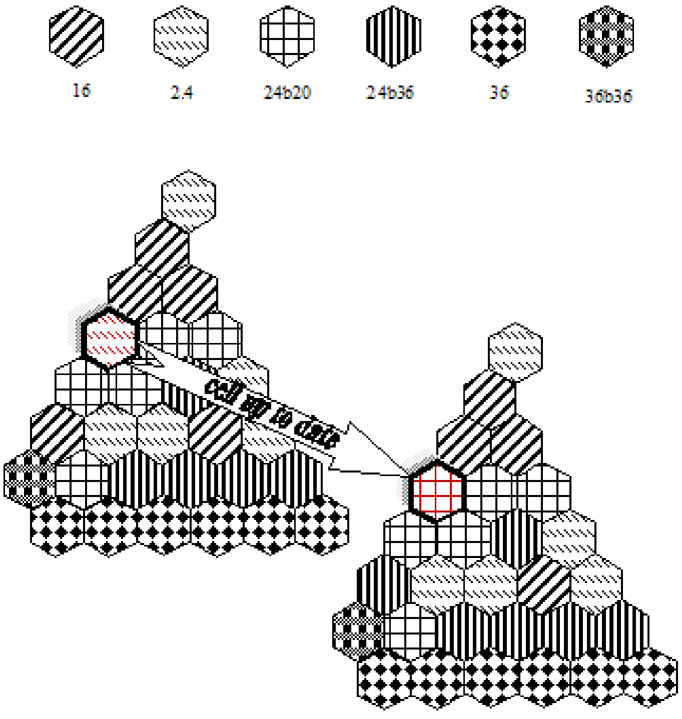
- CA structure
- Neutronic calculation

CA structure subroutine contains structure of Cellular Automata, kind and state of each cell. Besides this section perform the random selection of each cell and it's updating.

Neutronic calculation subroutine calls WIMS and CITATION codes for core calculation and finding the power distribution over reactor core.

Figures 20 to 23 show the evolution of the calculation by CA program from initial state to final state, where it is converged to an acceptable solution. In this way reaching to the global minimum is assured in acceptable time. As it is expected the solution found by proposed method is near to global minimum of energy function.

Evolution of two considered parameters in fuel management optimization procedure of VVER-1000 reactor core in proposed method had been shown in Figure 24.

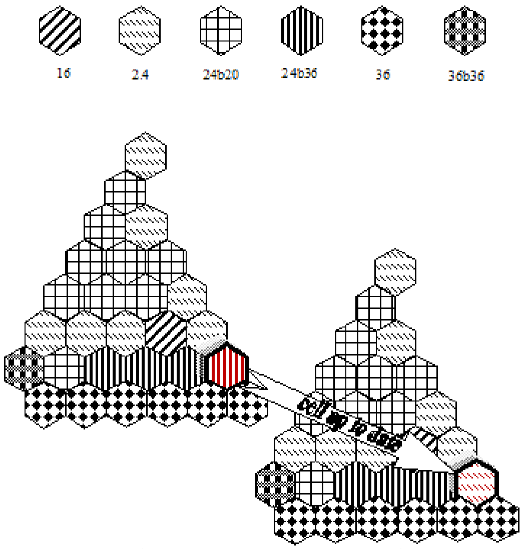


(a) core shape before displacing

(b) core shape after displacing

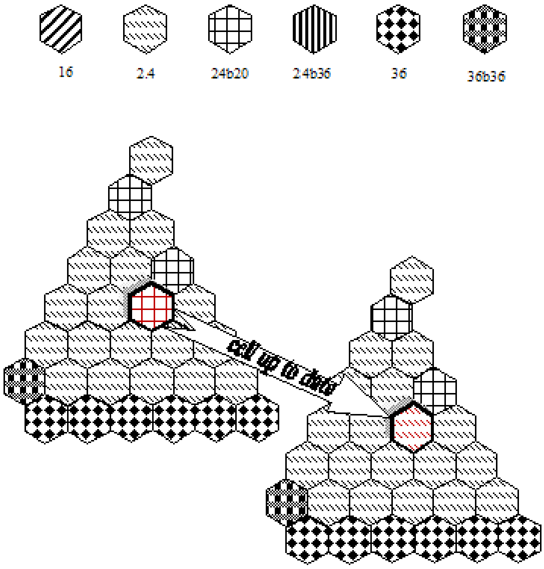
Figure 20. CA process for first loading of Bushehr NPP – first iteration (F_R =1.4966 and K_{ef} =1.1625)

If the F_R and K_{ef} parameters of the end configuration that given by proposed method, Figure 24 (F_R =1.3130 and K_{ef} = 1.2304), is compared to the energy of the configuration proposed by VVER Russian Designer (Figure 18), which are F_R =1.39 and K_{ef} =1.1577, we will conclude that the solution found by the proposed method is closer to optimum.



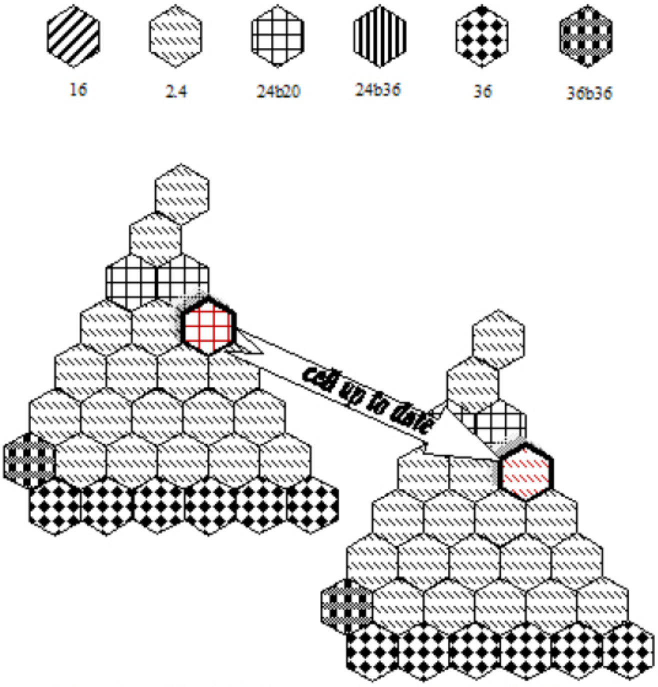
(a) core shape before displacing (b) core shape after displacing

Figure 21. CA process for first loading of Bushehr NPP –12th iteration (F_R =1.5929 and K_{ef} =1.1777)



(a) core shape before displacing (b) core shape after displacing

Figure 22. CA process for first loading of Bushehr NPP – 29th iteration (F_R =1.3069 and K_{ef} =1.2302)



(a) core shape before displacing

(b) core shape after displacing

Figure 23. CA process for first loading of Bushehr NPP – last iteration (F_R =1.3130 and K_{ef} =1.2304)

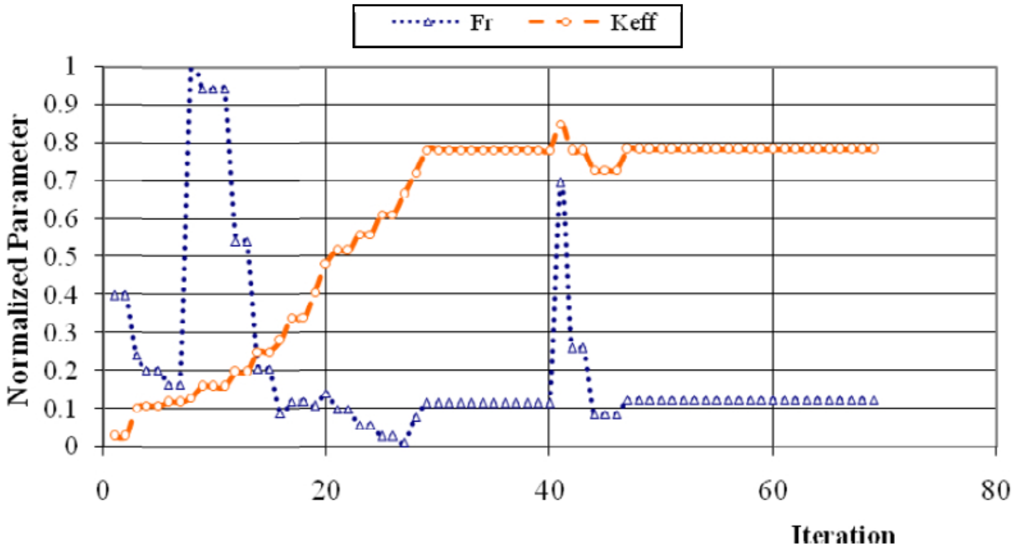


Figure 24. Evolution of F_R and K_{ef} parameters in the CA procedure for first loading of Bushehr NPP

As shown in table 3 calculated result by CA method for loading optimization as compared to the configuration, which proposed by VVER-1000, has lower power peaking factor and higher initial excess reactivity; therefore this method has good reliability for finding the optimum core configuration.

Needless to say that by improving the objective function and exceeding the parameters that have important role in fuel management and loading pattern the final result improved.

Table 3. Comparison of results

Method	F_R	K_{ef}	No. of Iterations
CA	1.31	1.2304	47
VVER-1000	1.39	1.1577	-

5. Conclusion

In this study, we introduce two new conceptual methods for optimization using cellular Automata. These methods were applied to loading pattern optimizations of VVER-1000 reactor and compared with the configuration, which proposed by designer. The final results, which have obtained in this way, are closer to the optimum solution.

In this section to demonstrate the advantages of proposed methods Hopfield Neural Network with Simulated Annealing algorithm was used for validating proposed methods.

Neural network models, with massively parallel structures, accompanied by simulated annealing method are powerful enough to find the best solution in reasonable time. Hopfield neural network operates as a local minimum searching algorithm; for improving obtained result from neural network, simulated annealing is used. Simulated annealing, because of its stochastic nature, can escape the result of Hopfield neural network from local minimum and guide to global minimum.

Based on mentioned method in Reference [18] as LONSA, fuel management optimization was performed using HNN with SA. The objective function for this method was defined as Eq. 28.

$$E\,4 = D/2\left(F_R - 1\right)^2 - H\ /2(K_{ef} - 1)^2 \tag{28}$$

where D=1000 and H=1000.

The other details information for LONSA is described in belonging paper that is cited in reference.

The results of this problem that was obtained from Hopfield neural network and simulated annealing algorithm [3] are shown in table 4 and Figures 25 & 26.

Table 4. Comparison of results

Method	F_R	K_{ef}	No. of Iterations
CASA	1.30	1.1625	77
Hopfield N.N.	1.35	1.1606	376
Hopfield & SA	1.32	1.1620	409
CA	1.31	1.2304	47
VVER-1000	1.39	1.1577	-

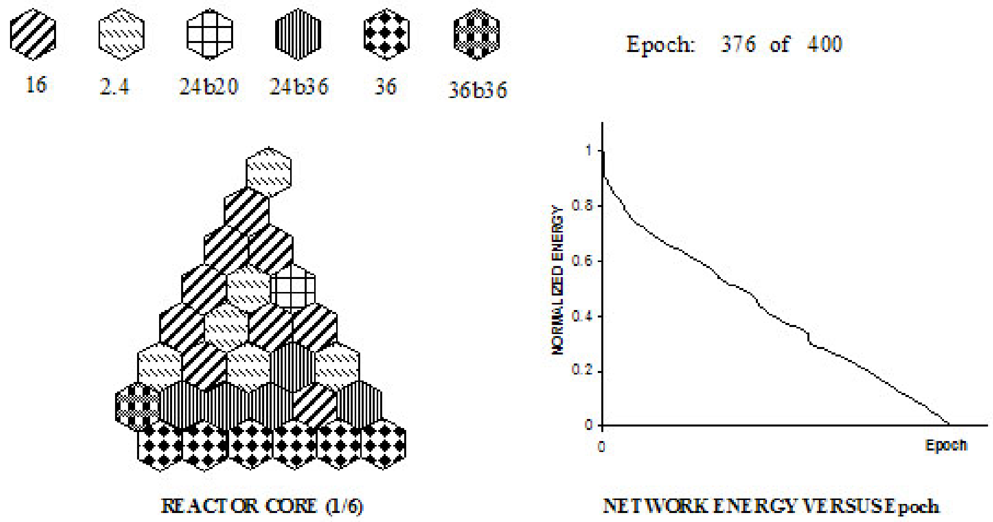


Figure 25. final configuration and evolution of energy that obtained using Hopfield Neural network for first loading of Bushehr NPP – (F_R = 1.35 & K_{ef} = 1.1606)

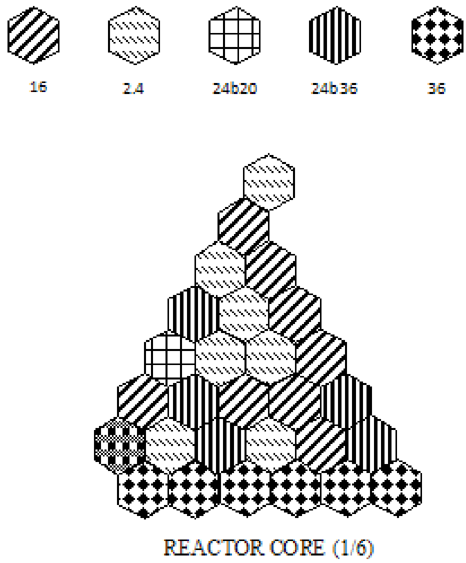


Figure 26. Final configuration that obtained using Hopfield neural network with simulated annealing for first loading of Bushehr NPP – (F_R = 1.32 & K_{ef} = 1.1620)

Because of the stochastic nature of mentioned methods, the results are very dependent on this procedure. In Neural network, the neuron selecting procedure for update is stochastic, and therefore the results are affected from this procedure. The simulated annealing procedure has the same nature, too. The replacing procedure obeys from the defined stochastic procedure. Also, the proposed method based on CA accompanied by SA has the stochastic nature. Cell selection, cell replacement and some other procedure are stochastic, and therefore the results are very dependent on those procedures.

In the methods that are based on stochastic procedure, the results comparison isn't a reliable method to validation. In this study, for checking the methods, we compare the quality of the set of results that are obtained from the subjected methods. Figure 27 & 28 show the quality of the performed optimization by HNN+SA and CA+SA in the 50 different iterations. The minimization of the F_R parameter is considered in the analytic assessment. As observed from the obtained results, the probability of finding the better results in the CA+SA method are very higher in comparing with HNN+SA method, thus the proposed method have good reliability in comparing with the other current methods.

In the second proposed methods, optimization by CA, the stochastic procedure has no effects on the procedure or maybe very week effects. But this method has a main part that can be very difficult to implement. Defining the cost function based on transition rule and local information whereas the cell updating lead to improve the cost function is the important stage of this method that need to analytical analyzing the system.

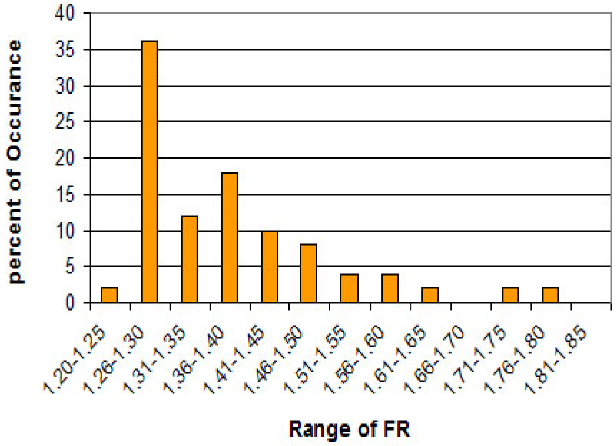


Figure 27. The results distribution of HNN+SA method in 50 iterations

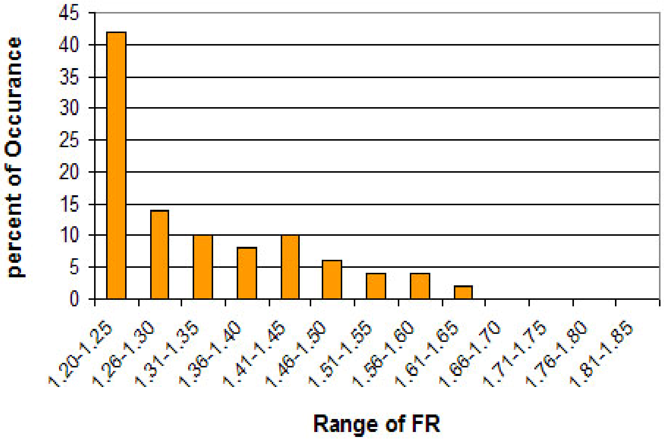


Figure 28. The results distribution of CA+SA method in 50 iterations

Consequently, based on mentioned cases, by comparing the proposed methods with neural network and simulated annealing that are classical methods in optimization, the reliability, velocity and quality of solution were checked, and the benefits of these methods have been demonstrated.

REFERENCES

[1] Neumann، J. V. Burks, & Urbana, A. W. (1966). *"The Theory of Self-Reproducing Automata "*، *IL*: University of Illinois Press، Ed.

[2] Bar-Yam, Y. (1997). *"Dynamic of Complex Systems"*، Addison-Wesley، June.

[3] Tatting, B., Gürdal, Z., 2000. Cellular automata for design of two-dimensional continuum structures. In: 8th AIAA/USAF/NASA/ISSMO *Symposium on Multidisciplinary Analysis and Optimization*

[4] Burks, W. (1970). *Essays on Cellular Automata*. University of Illinois Press, Urbana, Illinois.

[5] Gardner, M. (1970). The fantastic combinations of john conway's new solitaire game life. *Scientific American*, *223(4)*, 120-123.

[6] Wolfram, S. (1986). Theory and Application of Cellular Automata.*World Scientific*.

[7] Smith, M. A. (1994). Cellular Automata Methods in Mathematical Physics. Ph.D. *Thesis at the Massachusetts Institute Of Technology*.

[8] Rao, Singiresu, S. (1996). *Engineering Optimization Theory and Practice*, Third Edition, School of Mechanical Engineering, Purdue University West Lafayette, Indiana John Wiley & Sons, Inc.

[9] Fadaei, A. H. (2009). *Optimization of Fuel Management in VVER-1000 Reactor Core by Neural Networks & Cellular Automata,* Ph.D. Thesis, Amir kabir university of Technology.

[10] Aarts, E. & Korst, J. (1989). Simulated Annealing and Boltzmann Machines – A Stochastic Approach to Combinatorial Optimization and Neural Computing. John Wiley and Sons, New York.

[11] Van Laarhoveen, P. J. M. & Aarts, E. H. L. (1987). *Simulated Annealing: Theory and Applications.*. D. Reidel Publishing Company, Dordrecht, Holland.

[12] Stevent, J. G., Smith, K. S., Rempe, K. R. & Downar, T. J. (1995). Optimization of Pressurized Water Reactor Shuffling by Simulated Annealing with Heuristics. *Nucl.Sci. Eng.*, *Vol. 121*, No 1, 67.

[13] Kropaczek, D. J. & Turinsky, P. J. (1991). In-Core Nuclear Fuel Management for Pressurized Water Reactors *Utilizing Simulated Annealing., Nucl. Technol.*, Vol. 95, No. 1, 9.

[14] Fadaei, A. H. & Setayeshi, S. (2009). "An Optimization Method Based On Combination of Cellular Automata & Simulated Annealing for VVER-1000 NPP Loading Pattern". *Nuclear Engineering and Design* (*Elsevier Journal*), *239*, 2800–2808.

[15] Fadaei, A. H. & Setayeshi. S. (2009). "Control rod worth calculation for VVER-1000 nuclear reactor using WIMS and CITATION codes". *Progress In Nuclear Energy* (*Elsevier Journal*), *Vol 51*, No. 1, January.

[16] Faghihi, F., Fadaie, A. H. & Sayareh, R. (2007). Reactivity coefficients simulation of the Iranian VVER-1000 nuclear reactor using WIMS and CITATION codes., Progress *In Nuclear Energy*, (Elsevier Journal), *vol.49*, issue 1, January, 68-78.

[17] Fadaei, A. H. & Setayeshi, S. (2009). A New Optimization Method Based On Cellular Automata for VVER-1000 Nuclear Reactor Loading Pattern. *Annals of Nuclear Energy* (*Elsevier Journal*), 36 (May (5)), 659-667.

[18] Fadaei, A. H. & Setayeshi. S. (2008). "LONSA, As A Tool for Loading Pattern Optimization Using Synergy of Neural Network & Simulated Annealing for VVER-1000". *Annals of Nuclear Energy* (*Elsevier Journal*), *Vol 35*, N. 10, October, 1968-1973.

In: Cellular Automata
Editor: Thomas M. Li, pp.39-57

ISBN: 978-1-61761-592-4

Chapter 2

Modeling Drug Release using Cellular Automata: Evolution and Trends in Pharmaceutical Sciences

Nicolas Bertrand and Grégoire Leclair[*]
Faculty of Pharmacy, University of Montreal, Montreal, Canada

Abstract

Pharmaceutical science is a multidisciplinary field, in which fundamental and applied sciences unite to study all aspects related to drugs and pharmaceutical products. Amongst the specialities of this field, drug delivery focuses on altering the method of administration of therapeutic molecules to improve their effectiveness and safety. Various strategies exist to limit the toxicity or decrease the frequency of administration of an active ingredient. For example, polymeric biodegradable matrices are able to sustain and control the release of the drug. These systems are drug-reservoirs which are usually implanted subcutaneously or intramuscularly to discharge their load over a few weeks to a few months. The polymer implants may be fashioned in all kinds of shape: from films or larger blocks to micro- or nanosized particles.

The engineering of these formulations is complex and requires the combined expertise of polymer science, physical chemistry, and biology. In this context, cellular automata have emerged as interesting tools to model the various phenomena involved in the successful design and performance of these drug delivery systems. The following mini-review proposes a look at the different models available to depict the phenomena implicated in the release of drugs from biodegradable polymeric matrices.

Chronologically, the first aspect approached with the use of cellular automata was polymer erosion. The present work describes the various models proposed over the years, and emphasizes on the differences between them, from the probabilistic two-dimensional systems of the 1990's to the more comprehensive models proposed recently. In the second part of the manuscript, the authors focus on drug diffusion in the matrix, which is a phenomenon elegantly modeled by cellular automata. The various techniques used to describe molecular Brownian motion through confined space are also presented. Special

[*] Corresponding author: Email: gregoire.leclair@umontreal.ca

these aspects on drug release, and to demonstrate that adequate modeling can achieve better understanding of these complex pharmaceutical systems.

Throughout this work, the authors wish to present a critical view of the objectives achieved through the use of cellular automata as model and bring to light the impending improvements which remain to be accomplished to consolidate their role in drug delivery science.

1. Introduction

Cellular automaton (CA) models were first introduced by Von Neumann in 1948 as discrete dynamic models to simulate complex systems [1]. CA models are constituted of three basic components: a grid, numerous cells and local intuitive rules. The grid determines the spatial relationship between the cells. Cells are defined by their discrete state. Local rules dictate how cell can evolve according to their present state and the states of their neighbours. Local rules are iteratively applied for each cell and the grid is updated accordingly. A CA model in its simplest form was popularized in 1970 when Conway's Game of Life was presented in the Scientific American popular science journal [2].

The first application of a CA model in pharmaceutical sciences dates from 1990 by Zygourakis [3]. This author reported the use of a CA model to simulate the release of drug from a biodegradable polymeric drug device. Since then, several models have been defined to simulate complex systems. Table 1 provides a summary of all the reviewed models. Zygourakis' first model and most of the CA models developed since then share a few common points: (1) The CA grid is usually Cartesian and is defined in 2 or 3 dimensions; (2) A cell represents a defined region of quantifiable size; (3) Cell state is discretized and depicts the content of this spatial region (polymer, solvent, drug, etc.); and (4) intuitive rules are defined based on expected or known physical principles. The possibility to discretize and simplify equations figures among the advantages offered by this modeling design over the traditional numerical models. This feature also guarantees the predictive nature of CAs compared to models in which the data is fitted over experimental observations. Furthermore, the spatial representation used in CAs circumvents most of the restrictive shape assumptions governing models which use equations. It also allows the caption of images and movies throughout the simulations, which often broaden the understanding of the system under study.

Multiple variations of CA models exist. For instance, the Game of Life is based on a deterministic model were cells are defined by a unique nature at each iteration. This deterministic model is cell-type based. The nature of a cell at a given iteration is determined by its nature and the nature of the neighbouring cells at all previous iterations. In this type of model, the illusion of object movement, reduction or expansion is created by defining appropriate rules regulating the nature of each cell. As pharmaceutical concepts are often difficult to define deterministically, a reduced number of stochastic rules are often utilized instead of multiple deterministic rules. This allows the simulation of more complex processes using fewer iterations and a coarser simulation space.

Table 1. List of reviewed CA models published between 1990 and 2009

Author *Journal* [Ref]	Year	Erosion	Cell distribution	Neighbours	Diffusion	Initial porosity	Drug release	Dimensions
Zygourakis								
Chemical Engineering Science [3]	1990	When LE < Threshold LE ↓ according to probabilistic or deterministic rules	Initial state: Bernouilli trials	-	-	Randomly distributed circles of specific diameter	Drug cells eroded	2D
Biomaterials [4]	1996	When LE < Threshold	Initial state: Bernouilli trials LE: 1st order Erlang	Von Neumann LE ↓ linearly with *N*	-	Randomly distributed circles of specific diameter	-	2D
Göpferich								
Macromol [5]	1993	When LE < Threshold LE start to ↓ when neighbour eroded	Cristallinity: Bernouilli trial LE: 1st order Erlang	Eroded neighbour Y or N ?	-	-	-	2D
AIChE Journal [6]	1995	When LE < Threshold AND in contact with surface	Cristallinity: Bernouilli trial LE: 1st order Erlang	Eroded neighbour Y or N ?	-	-	Amorphous polymer cells eroded	2D: Rotationally symmetric
J Control Release [7]	1995	When LE < Threshold	Cristallinity: Bernouilli trial LE: 1st order Erlang	-	Numerical resolution of Fick's law	None	Monomer release	2D
Macromol [8]	1997	When LE < Threshold AND in contact with surface	Cristallinity: Bernouilli trial LE: 1st order Erlang	Contact with surface Y or N.	-	None	-	2D

Table 1. (Continued)

Author *Journal*[Ref]	Year	Erosion	Cell distribution	Neighbours	Diffusion	Initial porosity	Drug release	Dimensions
Barat								
Sim Model: Practice & Theory [14]	2006	No erosion, only excipient dissolution	Initial state: Bernouilli trials	No effect on erosion	Random walk (compartimental)	None	Drug cells diffusing outside of limits	2D
J Pharm Biomed Anal [15]	2008	When LE < Threshold	LE: 1st order Erlang	No effect on erosion	Random walk	Method not specified	Drug cells diffusing outside of limits	3D
Theory Biosci [16]	2008	When LE < Threshold	LE: 1st order Erlang	No effect on erosion	Random walk	Method not specified	Drug cells diffusing outside of limits	3D
Vlugt-Wensink								
J Control Release [17]	2006	When time > LE	In clusters	-	Random walk	None	Drug cells diffusing outside of limits	3D
Hildgen								
Int J Pharm [18]	2007	When random number > probability of non-erosion (P_{NE})	Log normal distribution and modified log normal	Von Neumann P_{NE} ↓exponentially with *N*	Random walk	Pores generated by random walk	Drug cells diffusing outside of limits	3D
Yu								
Sim Model: Practice & Theory [19]	2008	When LE < Threshold	LE: 1st order Erlang	Eroded neighbour Y or N ?	-	None	Number of Drug cells eroded	3D
Laaksonen								
Biomaterials [20]	2009	When random number < probability of erosion (P_E)	Initial state: Bernouilli trials	Von Neumann P_E ↑ linearly with *N*	Random walk (compartimental)	None	Drug cells diffusing outside of limits	2D
Int J Pharm [21]	2009	Polymer density ↓ randomly & with swelling When polymer density reaches 0	Initial state: Bernouilli trials	Von Neumann P_E ↑ linearly with *N*	Random walk (compartimental)	None	Drug cells diffusing outside of limits	2D

LE: Life expectancy; *N*: number of water-containing neighbouring cells; P_E: Probability of erosion, P_{NE}: Probability of non-erosion (1-P_E), Y: Yes, N: No.

Another variation of the original CA model includes compartment-like models. Instead of focussing on cell types, these latter models focus on the amount of a given object present in a given cell. In some cases this quantity is discretized, in other cases it is a real number value. Rules are defined to regulate the mass transfer from one cell to the others. Therefore these CA models are in some ways similar to pharmacokinetics compartmental models: cells behave like compartments and local rules are similar to transfer micro-constants. This approach is convenient to simulate diffusion as it models easily the concepts associated with concentration gradients.

Finally some models are hybrid: CA rules are first used to simulate a complex process such as dissolution and erosion. The different dependant variables of the model (for example the diffusion coefficient) are then evaluated at each iteration. These dependant variables can then be utilized to resolve complex differential equations such as Fick's diffusion laws.

Since 1990, various CA models have been developed to address three pharmaceutical problems: (1) degradation, erosion and dissolution of biodegradable polymers; (2) dissolution of drug substances; and (3) diffusion of drugs out of a polymeric device. The next section proposes to provide an historical review of these models. The third and fourth sections summarize the sequence of events resulting to drug release from a polymeric device and how these events can be modeled using CAs. Finally, a critical review on the calibration as well as the predictive value of these CA models is provided in the fifth section of this chapter.

2. Historical Review

In 1990, Zygourakis defined a simple but efficient CA model [3]. This 2D model comprised 2048 x 2048 cells. Four cell states were defined: (1) polymer, (2) drug, (3) solvent and (4) pore void. Cubic tablets were modeled with different drug loading and porosity. Porosity and drug circle of known size were distributed over a lattice of polymer. The model evolved according to two rules: (1) a pore cell was filled with solvent if it had at least one solvent neighbour; (2) a polymer or drug cell was replaced with solvent after a deterministically or stochastically computed number of iterations if it had at least one solvent neighbour. A drug release event was accounted every time a drug cell was replaced by solvent. This model was good to evaluate drug release from a surface erosion controlled drug delivery device. The model could simulate the penetration of solvent in the device through porosity. However, it could not simulate the diffusion of drug throughout the device. The model simply assumed that as soon as the drug was dissolved, it was released.

In 1996, Zygourakis presented a slightly improved CA model based on his previous model [4]. A 250 x 250 cell lattice was filled with different solid components, porosity void or solvent. Each solid component (drug or polymer) cell in the lattice was assigned a life expectancy which could either be constant or could vary according to a Poisson distribution. At every iteration, the life expectancy of a solid cell was reduced by the number of neighbouring solvent cells. Once life expectancy reached zero, the cell was replaced with solvent. For drug cells, this counted as a release event. The benefit of this model in comparison to the previous one was to allow the modeling of ternary and even more complex systems; however, only binary systems were evaluated. This model also took into account the number of neighbouring solvent cells to determine the dissolution time of a given solid cell.

In the meantime (1993), Göpferich and Langer developed another CA model to simulate the erosion of a cylindrical polymeric device [5]. Taking advantage of the symmetrical properties of cylinders, they modelled only half of a cross-section of the device using a 100 x 100 cell simulation space. Three cell states were defined: (1) slow-eroding crystalline polymer; (2) fast-eroding amorphous polymer; and (3) solvent. Randomly distributed binary systems of known crystalline-to-amorphous ratio could be modeled. This model was based on a single rule: once a polymer-containing cell was exposed to a solvent cell, it was replaced by solvent after a stochastically determined number of iteration (life expectancy). Life expectancy was based on a first-order Erlang probability function specific for each component.

In 1995, this model was fine tuned to simulate drug release from coated and non-coated cylinder devices [6]. To quantify drug release, it was necessary to take into account that the volume of a cylinder is proportional to the squared radius. Non-constant cells of rectangular geometry were used for this model. The x-length of these cells was a function of its distance from the center of the device. This simplification allowed a more accurate simulation of a 3D process using a 2D grid. It should be noted that drug was not a cell state in these two latter models; it was simply assumed that drug was uniformly distributed within the polymer and would be released at the same rate as polymer would erode.

Also in 1995, a significant addition to this model was made which took into account the diffusion of monomers out of the polymeric matrix [7]. A similar CA model was designed to simulate the morphological changes during surface erosion of a polymeric matrix. This allowed the calculation of porosity distribution throughout the matrix. Eroded polymer cells were assumed to be replaced by their equivalent monomer. Diffusion was not simulated using a CA model, but rather calculated according to Fick's first law, using the porosity calculated from the CA model to determine the effective diffusion coefficient. This strategy integrated two radically different approaches into a single hybrid model.

In parallel, Göpferich widened his perspective of erosion in 1997 to examine bulk polymer erosion [8]. The life expectancy model developed earlier [5] was modified by discriminating between polymer degradation (breakage of chemical bonds) and polymer erosion (disappearance of polymer cells). Hence, the start of the life expectancy decay of polymer cells was synchronous; cells with expired life expectancies were considered as degraded, but erosion only occurred when degraded cells were in contact with the surface (*via* contiguous solvent cells). This feature assured homogenous degradation throughout the polymer bulk, and was subsequently used in other work to explain different experimental observations [9, 10]. However, by design, the model lost the ability to portray surface erosion.

In 1994, Kier and Cheng published the details of an innovative CA model of water. This model was designed to simulate molecular-scale events. The 2D simulation space was a boundless region of 40 x 40 cells representing the surface of a torus. Only two cell states were defined: (1) empty and (2) water molecule. Water molecules would move within the simulation space according to three stochastic rules: (1) a water molecule would freely move to one of the neighbouring cell if all its first level and second level neighbourhood were empty; (2) two or more vicinal water molecules formed a cluster; clusters were static, but molecules could detach from a cluster according to a breaking probability rule; and finally (3) if a water molecule had another water molecule in its second level neighbourhood, a joining probability would determine if it would interact with it and form a cluster. This relatively simple single component model was surprisingly efficient to calculate different physical properties of water such as hydrogen bounding, viscosity and vapour pressure.

The basic CA model of water has been fine tuned by this research group [11-13]. By using similar rules between molecules of different types, many more physical processes can be simulated: solubility, oil-water partitioning, dissolution, micelle formation, percolation and diffusion. This molecular dynamic approach is an elegant way of resolving key physical problems which are pertinent to pharmaceutical sciences. On the other hand, this molecular dynamics strategy has not been used as of yet to design CA models of drug release from drug delivery devices.

In 2006, Barat published a Monte Carlo model of drug dissolution [14]. This model was used to simulate the dissolution of two-component compacts in a standard experimental context (United States Pharmacopeia type 2 dissolution apparatus). By design, this model had some CA flavour in its definition of the simulation space, a 2D lattice of cells. Defined cell types included drug, excipient, leaking drug, leaking excipient, static layer of solvent and bulk solvent. In addition to cell type, each cell could contain a defined number of excipients and drug particles representing the concentration of these components within the cell. Drug and excipient particles migrated from leaking cells to static layer cells, between static layer cells and eventually from static layer cells to bulk solvent cells. Stochastic rules based on diffusion equations were used to quantify the number of particles migrating from on cell to another.

The latter model of drug dissolution was complexified in 2008 [15, 16]. Release of proteins (*i.e.* drug) from a polymeric microsphere was simulated using a compartment-like CA model. A 3D simulation space was used. On the one hand, a polymeric microsphere was modeled by filling the simulation space with polymeric cells, porosity void space and also protein cells. From this polymeric sphere model, erosion was simulated according to stochastic rules as described previously by Göpferich [5]. Alternatively, diffusion of dissolved protein through the porosity was modeled using random walk rules. A release event was calculated when a dissolved protein escaped the region of the simulated space occupied by the polymeric microsphere.

Meanwhile in 2006, another group published a CA-like, stochastic model [17]. In this work, polymer erosion of a definite number of neighbouring cells was used as a requisite binary condition to allow diffusion of a macromolecular drug; diffusion was only possible if pores were large enough. For this reason, clusters of a definite number of cells were attributed randomly determined life-expectancies instead of individual cells. The number of cells in a cluster was a function of both the polymer cross-linking density and protein hydrodynamic diameter. The 50 x50 x 50, three-dimensional lattice was then used to model drug diffusion through contiguous empty pores. This model is distinctive because the usage of clusters to model one phenomenon allows representation of different size-scales for erosion and diffusion.

In 2007, Hildgen published a CA model inspired by the pioneer works described above [18]. In this work, a three-dimensional environment (300 x 300 x 300 cells) was defined, and cells were attributed the specific states of polymer, drug and porosity. Erosion was modeled in a stochastic perspective by attributing a non-erosion probability (the complementary probability of an erosion event) for each cell. For polymer cells, this probability decreased exponentially with each solvent-containing neighbour. Different polymer cells were attributed distinct non-erosion probabilities according to a log normal distribution. In order to realistically depict the drug delivery devices, particular attention was given to the representation of porosity. This feature significantly improved modeling of drug diffusion by a random walk. Drug was considered released when it escaped the volume defined by the polymer cells.

Similarly, a 3D extension of Göpferich's CA model was also proposed by Yu *et al.* in 2008 to simulate drug release from erosion-controlled polymeric devices [19].

In 2009, Laaksonen proposed of CA model to simulate drug release from diffusion-controlled polymeric devices [20]. The 2D simulation space (125 x 125 cells) was populated with six different types of cells: (1) water; (2) solid drug; (3) dissolved drug; (4) solid polymer; (5) wet polymer; and (6) wet polymer with dissolved drug. Stochastic rules were defined to control the probability that a solid cell would change into a wet cell upon contact with any type of wet cells. Another stochastic rule determined the probability of erosion of a wet polymer cell. To address drug diffusion, all cells containing drug were defined as compartments containing a finite amount of drug. The drug dissolved in wet cells would be submitted to a random walk through any neighbouring wet cells.

In another recent study, the same group modified their model by introducing polymer-swelling as a dominant parameter in the stochastic CA [21]. In this model, the polymer cells were considered as compartments and both the erosion probability and swelling probability were dependent on the polymer "density" of each polymer cell. Swelling decreased density of a given cell by augmenting the density of one of its solvent-containing neighbour, while erosion decreased the density without affecting neighbouring cells.

3. MODELING MATRIX EROSION

Many physical phenomena are implicated in drug release from polymeric drug delivery system (PDDS), and throughout the last twenty years, scientists have put efforts into modeling them using CAs. The following section will focus on erosion which represents the loss of mass from a polymer matrix [22]. The objective will be to put forward how the CAs in literature convene with the current experimentally-obtained understanding of this complex phenomenon.

Firstly, as pharmaceutical sciences regroup an array of specialities, an important amount of experimental observations are available when initiating the design of a model. Most of the time, the initial configuration of the model decides which parameters will be addressed and which will be left out. Hence, the initial representation of the PDDS in the CA will be discussed in detail. Secondly, as shown in Figure 1, for an intricate system of entangled polymer chains, polymer erosion results from three distinct physical events: (1) water ingress in the polymeric structure; (2) polymer degradation, involving the breaking (*i.e.* hydrolysis) of chemical bonds and the decrease in polymer chain length and (3) the diffusion of the polymer chains or oligo/monomers out of the polymer matrix. The integration of these three events in CAs is discussed in further detail.

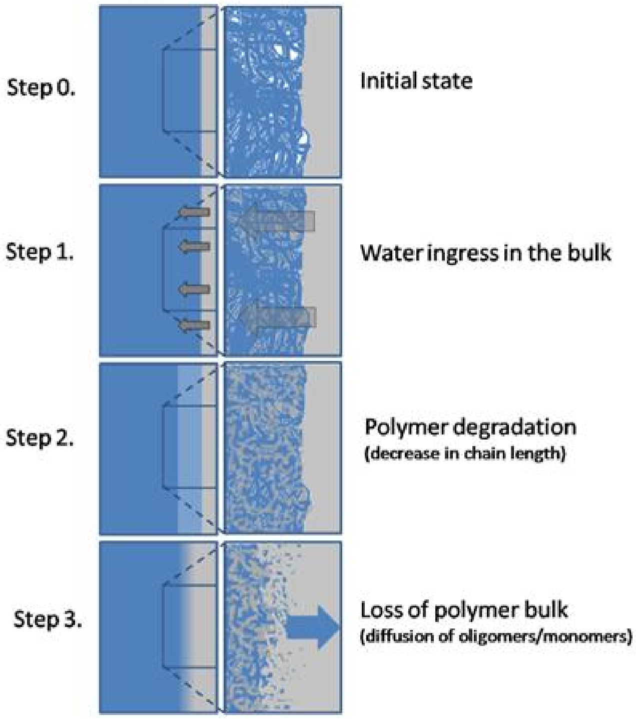

Step 0.
Initial state
Step 1.
Water ingress in the bulk
Step 2.
Polymer degradation
(decrease in chain length)
Step 3.
Loss of polymer bulk
(diffusion of oligomers/monomers)

Figure 1. The four steps leading to polymeric erosion

3.1. Describing the Primary State of the Matrix

The principal part of the bulk of a PDDS consists of polymer. Polymers are high molecular-weight molecules composed of covalently linked repeating units (monomers). The polymer matrix is held together by the entanglement of the chains and hydrophobic interactions between molecules. The cohesive forces of the bulk depend on the type of polymer (*i.e.* the type of monomers), the length of the chains (*i.e.* molecular weight of the polymer), and the crystallinity of the bulk [23, 24]. This last parameter is defined by the amount of local organization between the intra- and intermolecular chemical groups. In a crystalline polymer, the chains (or part of the chains) are ordered in a very tight fashion and the forces holding the matrix together are hard to overcome. In an amorphous polymer, no local organization is discernible, and less energy is required to break intermolecular interactions. Experimentally, effects of polymer type, molecular weight and crystallinity on erosion were thoroughly studied in various works [25-28].

These parameters were amongst the first to be integrated in CA, and still figure as the core of the known erosion models. In his work, Zygourakis defined polymer cells with different erosion rates by attributing life expectations for each cell, with long life expectations corresponding to slow erosion rates [4]. This trend was refined by Göpferich and Langer [5] who assigned “crystalline” and “amorphous” types to each polymer pixels. The cells of each type had life expectations distributed according to two distinct Erlang distributions. Similar approaches were used in more recent models [14, 19]. In others, erosion was determined stochastically by distributing erosion probabilities instead of life expectations [18, 20, 21]. In most of these works, the distribution of each cell type during matrix generation is made arbitrarily for each pixel without consideration for the other cells in its vicinity. This stochastic conception might deviate from the more organized distribution of crystalline domains often observed experimentally [24, 26, 27].

Another important component of all PDDS is the active ingredient. In reservoir-type controlled release systems, the active agent is physically immobilized in a water insoluble polymeric scaffold that delays its contact with the aqueous environment. To date, the systems designed comprise of various types of molecules which extend from low-molecular weight drugs [18, 29] to macromolecular peptides and proteins [30]. In most of these formulations, the active ingredient is molecularly dispersed in the polymer and therefore a limited amount of drug-drug interactions occur. However, in certain systems, the drug is not homogeneously distributed [31-33]. Hence, the presence of drug-rich clusters or drug crystals requires the characterization of intrinsic parameters for the active ingredient, such as solubility (the maximum amount of drug which can be dissolved in solution), crystallinity (the strength of cohesive forces between drug molecules) and polymorphic state (a measure of cohesive forces, in non-crystalline states). In this type of PDDS, other formulation-specific considerations also exist, like drug-cluster size and shape.

For the reasons mentioned above, dissolution of the drug is rarely the rate-limiting step in models describing erosion-controlled drug release, and drug is usually distributed randomly in the matrix, as single pixels [20, 21]. However, Zygourakis [4] and Hildgen [18] have proposed models which can distribute drug as spherical clusters of specific diameters as if they were crystals or a hydrophilic drug encapsulated by the double-emulsion method. Although the influence of this feature on release rates in simple erosion models is minor for realistic drug loadings (< 20-30%) [4], it becomes interesting when the models combines polymer erosion and drug diffusion (see Section 4 below) [18].

It is noteworthy that as pharmaceutical research evolves, other formulation-dependent factors emerge. A few, such as the compatibility of the drug with the polymer [34] are system-specific and, until now, have not been implemented in any model. Others, like the porosity in the matrix, have only recently been examined experimentally [35-37], but were straightforwardly integrated in cellular automaton models since the beginnings [3, 4, 15, 18].

Porosity in the matrix was already featured in the first CAs. In Zygourakis' models, a specific quantity of circular pores was integrated in the drug-polymer blend as randomly distributed empty cells [3, 4]. Although this type of porosity was consistent with a surface eroding matrix, once again its effect on drug release rates was somehow limited (at least for realistically low drug loadings) [4]. In Göpferich's hybrid model, the role of porosity was acknowledged as a numerical variable which fluctuated according to polymer erosion and was integrated in the numerical resolution of Fick's diffusion law, but no initial porosity was attributed to the initial matrix [7]. In other models, the influence of randomly distributed porosity on drug release is studied quantitatively without detail on the pore shape [15, 16]. To the best of our knowledge, Hildgen was the first to consider randomly shaped pores. In this model, the pores were distributed in the form of connecting tunnels following a self-avoiding random walk throughout the matrix. This method was chosen amongst others because of its more realistic depiction of experimental features of porosity (surface area, surface fractal dimension, etc). The three-dimensional environment and integration of the drug diffusion process provided accurate burst-release and polymer bulk erosion modeling [18].

3.2. Step 1 of Polymer Erosion: Water Penetration in the Matrix

When PDDS are immerged in aqueous biological fluids their surface is wetted and capillary forces fill the pores connecting to the surface with water. The water uptake of matrices varies with the hydrophobicity of the surface and polymer bulk. The quantity, shape and size of pores connected to the external aqueous environment are also influential parameters. Experimentally, these factors are hard to study. For example, indirect measurement of the contact angle of an aqueous droplet on a polymer film could provide and estimation of the hydrophobicity of a PDDS surface [38], but could hardly supply a direct water uptake kinetic. Likewise, pore measurements obtained with various gas adsorption techniques only offer an approximation of the volume available to water: the high surface tensions of aqueous solutions would certainly prevent liquid penetration in the tightest pores [36, 37].

Nonetheless, it is the water taken up by the polymer mass that will lead to the further steps of drug release and dictate if the polymer undergoes surface or bulk erosion [8, 9, 39]. Since polymer hydrolysis occurs when the polymer chains are in contact with the aqueous media, percolation of the water through the polymer is essential in order to have degradation (and further erosion) inside the mass of polymer. If the water ingress in the matrix is slow or if the polymer is poorly resistant to hydrolysis, the surface of the polymer will erode much faster than the water front will penetrate inside the core, and surface erosion will occur [8, 9].

This concept was poorly understood when the first CAs were designed, and polymers were believed to undergo either surface or bulk erosion without considerations for the shape of the systems studied. For this reason, the early models [3-5, 7] were strictly designed to model one type of erosion. In the mid-nineties, Göpferich adapted his Monte-Carlo model to better represent polymer bulk erosion, by discriminating between polymer degradation and erosion [8]. After a randomly-determined life-expectancy, polymer underwent degradation (fell in the “degraded” state) but could not be eroded (changed to “eroded” state) until at least one of its neighbour contained water. Although pioneering at the time, this design was imperfect because water penetration in the matrix was depicted as a consequence of erosion rather than its cause. In recent models, the representation of water penetration inside the matrix was more realistically portrayed through accurate depiction of porosity [18] or polymer swelling [21]. Indeed, Laaksonen *et al.* [21] proposed an interesting model where penetration of water was achieved by determining a “wet polymer” state. When in contact with a water-containing neighbour, each polymer cell had a probability of absorbing a part of the surrounding milieu and thereby changed its state. Once in the wet form, the polymer underwent erosion or swelled in neighbouring cells, and the water was free to continue its progression throughout the matrix independently.

3.3. Step 2 of Polymer Erosion: Polymer Degradation

By definition, the chemical bounds forming a biodegradable polymer must be degradable under physiological conditions. Although polyesters are the most studied and better characterized class of biodegradable polymers [24], numerous other types of polymers have been used [22, 40]. Evidently, distinct types of chemical structures offer different polymer degradation kinetics. As the bonds are hydrolysed, the long polymeric chains are cleaved into shorter molecules, and the average molecular weight decreases accordingly. Various works have studied the way the molar weight of polymers declines in different conditions and evidenced the distinctions between polymer chain scission and loss of material in the medium [8, 9, 26, 27].

While trying to accurately model bulk erosion (*see* above), Göpferich was the first to recognize the distinction between the steps of degradation and erosion [8]. The decrease in molecular weight was determined to be a necessary condition for polymer erosion. In his simulations, although the decrease in the cell's life expectancy started simultaneously, it was established that polymer could not erode until it was in contact with the surface. Hence a latency period could occur between the life termination of a cell and its deletion from the matrix. This period corresponds to the time taken by the water to percolate throughout the matrix.

This refinement was lost in most contemporary models which frequently combined the degradation and erosion processes in the same mathematical parameter, whether it is life-expectancy or a probability of erosion. Separation of polymer degradation from erosion in two distinct events would permit more accurate representation of different experimental observations. Amongst other things, the influence of molecular weight on the erosion kinetics could be more straightforwardly depicted. Indeed, in a polymeric matrix, the degradation of one particular chemical bound is independent of its position in the macromolecule. However, the initial polymeric chain length and the location of the scission event will affect erosion kinetics by altering the cohesive forces and mobility of the degraded products. In current models, this detail remains restrained in a simple erosion parameter, but the spatial representation defined in CAs could be usefully employed to monitor this phenomenon.

Likewise, another aspect that could interestingly take advantage of the cell lattice in CAs is the characterization of the acidic autocatalysis occurring in the bulk of certain bioerodible polymers. During this process, the acid degradation products are trapped inside the polymeric network and catalyze the hydrolysis reaction occurring in the matrix. This results in heterogeneous degradation kinetics of the polymer according to its location in the PDDS and its proximity to the surface [41]. CAs could be exploited to define position-dependent degradation kinetics.

3.4. Step 3 of Polymer Erosion: Loss of Polymer Bulk

Most bioerodible matrices used as PDDS consist of enmeshed polymer chains held together by hydrophobic interactions. Upon decrease in the molecular weight of the molecules the cohesion forces maintaining the matrix together are reduced, and the chains can gradually disengage from the main bulk. This loss of polymer can occur either because degradation leads to the formation of soluble oligomers or monomers which are able to diffuse out of the mass, or because solid, insoluble portions of polymer detach from the main part and are cleared in the circulation.

Current erosion models do not ponder into all those details, and customarily erosion is represented by the "deletion" of a polymer cell and its replacement by a solvent pixel. Models use either deterministic life expectancies (often randomly distributed in the matrix) [3-6, 15, 16] or stochastic erosion events based on a series of erosion probabilities [18, 20, 21]. To represent the fact that polymer erosion occurs more rapidly when the polymer is surrounded in water, certain models shorten the time required for the erosion by weighing the number of solvent-containing neighbours (*N*). The most frequent method of doing so implies a linear relation between the time needed for erosion and the number of neighbours containing solvent [4, 19-21]. Another option proposed by Hildgen [18] adjusts the probability of undergoing erosion exponentially with the number or neighbours. To achieve this easily, the stochastic event is transformed in a non-erosion probability which exponentially decreases with *n*. The relevance of this feature has not yet been established, but certainly allows the erosion event to remain stochastic for maximum values of *n* (4 or 6 for Von Neumann neighbourhood in two and three-dimensions, respectively) in polymer cells with high intrinsic erosion probabilities.

4. Modeling Drug Diffusion

Similar to erosion, drug diffusion plays an important role during drug release from a polymeric matrix. On a molecular basis, Brownian motion of molecules in solution is rather uncomplicated as it involves a random movement in space. When a concentration gradient exists between two regions, this disordered movement results in a positive net flux of molecules from the concentrated to the more diluted environment. Diffusion or mass transfer is well described using Fick's first law (Equation 1). In this differential equation, the diffusive flux (J) is a function of the effective diffusion coefficient (D_{eff}) and the concentration gradient ($\partial C/\partial x$)

$$J = -D_{eff}\,\frac{\partial C}{\partial x} \tag{1}$$

This simple equation is useful for simple systems; however, it becomes impossible to find a solution for inhomogeneous and irregularly shaped 3D polymeric devices where both D_{eff} and C are functions of time and position. This problem was elegantly solved by Göpferich and Langer [7] using a hybrid model. The CA part of the model was used to simulate polymeric erosion. Erosion leaves pores in the polymeric device and porosity could easily be evaluated by counting the number of pore cells. The numerical part of the model used Fick's second law for one-dimensional diffusion through porous materials (Equation 2). As porosity $\varepsilon(x,t)$ could be evaluated using the CA part of the model, finding a numerical solution to this equation became possible.

$$\frac{\partial}{\partial t}C(x,t)\,\varepsilon(x,t) = \frac{\partial}{\partial x}D_{eff}(C)\,\varepsilon(x,t)\,\frac{\partial C(x,t)}{\partial x} \tag{2}$$

This complex approach, initially proposed to model monomer diffusion, has been adapted to address drug release [29]. However, although these models offer simulations which acutely correlate with experimentation, their numerical approach is somehow remote from direct CAs models, and has been extensively reviewed elsewhere [42, 43].

The more straightforward approach to model diffusion using CAs depicts movement of molecules throughout the lattice with random walks. Random walks are the resulting trajectory of a sequence of disordered steps, in all directions. The average distance strayed from the point of origin (R_{rw}) is proportional to the number of steps (n) and the size of the steps (δ) according to Equation 3 [44].

$$R_{rw} = \sqrt{n \times \delta} \tag{3}$$

When movement occurs in three dimensions, it can be easily correlated with the diffusion coefficient [44].

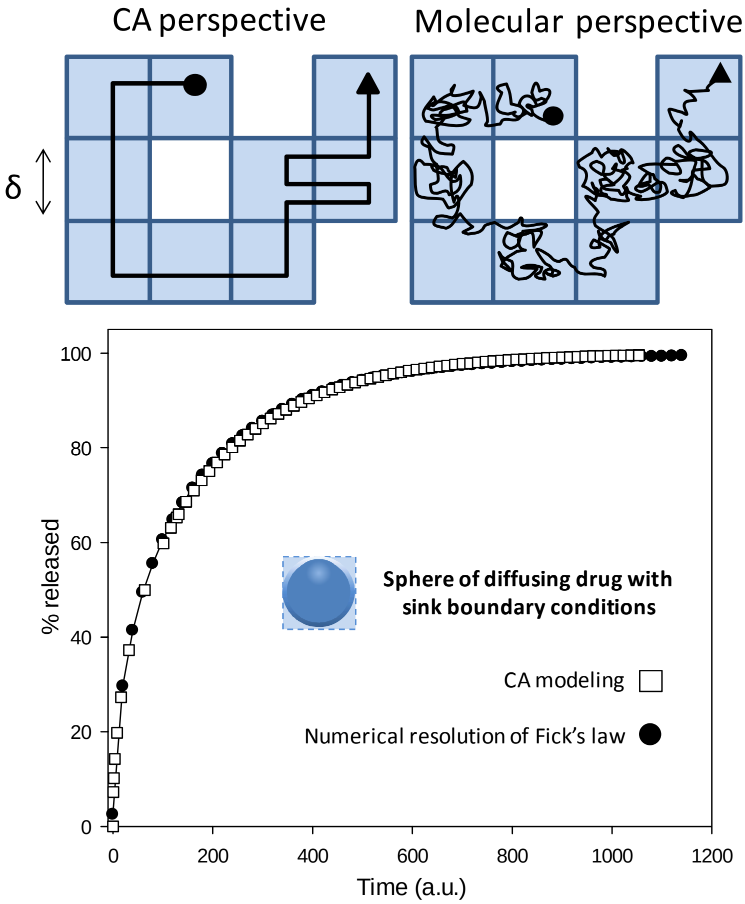


Figure 2. Random walk resolution of the diffusion process

In most CAs using random walks, one step occurs when the individual state of adjacent cells are exchanged with each other. Likewise, in compartmental-like models, the content from one cell is translated to an adjacent cell. The distance traveled in one step is dependent on the lattice scale and on the volume represented by each cell, but is usually well correlated with numerical resolutions of Fick's law (Figure 2) [45].

Subtle differences exist between models using random walks to model diffusion. For example, the models can be traditional CAs (with defined states) [18] or compartment-like [15, 16, 20, 21], have different type of neighbourhood (Von Neumann's or Moore's) [14, 16] or the diffusion rules can include a probability for a cell to remain in the same location [20, 21, 46]. Somehow, the models can also be deterministic [47]. The pertinence of all these features remain unknown and since diffusion is usually intertwined with other phenomenon (such as erosion or swelling), it is difficult to ponder which choices are more relevant.

5. Evaluating the Predictive Value of Models

Unlike certain theoretical specialities, the field of pharmaceutical sciences is supported by a number of easily-accessible experimental observations. For this reason, it is often appealing to compare the predictive results of a simulation with actual data obtained from experiments, and detect the strength and weaknesses of the modeling approach. However, since CAs offer discretization of time and space and the lattices are usually dimensionless, units obtained in simulations remain arbitrary. Hence, the scientists usually face problems to correlate *in silico* simulations with *in vitro* data.

In the models described above, numerous approaches have been used. The simpler method circumvents the problem by providing simulations results which are normalized and uncorrelated with any experimental sets of data [3, 4, 19, 20]. Other methods determine empirically certain constants or initial parameters used in the model. For example, Göpferich determine the life expectancies used in the model by experimentally measuring the erosion rates of polymer [5-8]. Although this approach becomes complex when different phenomena are intertwined in a model, certain authors succeed in establishing satisfactory parallels between experiments and simulations [21]. Other authors willingly abandon the initial dimensionless features of the CAs by integrating simple equations to convert one of the parameters used in the model into a known constant. For example, they might use the number of steps and step size in a random walk to obtain the diffusion coefficient [18]. Otherwise they simply define the units used in the model parameters [14-16]. It is noteworthy, that all these methods are usually time-consuming because a large number of simulations are required to tweak each of these parameters and obtain optimal results. Also, the major drawback of all these approaches is that they all rely on the *a priori* assumption that the model clearly represents reality, which is not necessarily the case.

In the same line of thoughts, one major imperfection of most models trying to relate their simulations with experimental observations is the lack of an objective assessment for the correlation between data. Indeed, most works rely on subjective visual observations to suggest that their model adequately represent experiments. This method seems inadequate when it is compared to other common types of model which use goodness-of-fit statistics. Although the same objective functions cannot be used with CAs which are predictive and not descriptive, Hildgen proposes an alternative with his model [18]. A weighted mean squared-difference between experimental observation and simulation is used to provide a relative error evaluation of the model and compare it to an objective index [48]. Widespread application of methods providing unbiased evaluation of the discrepancies between simulation and experiments could be useful to compare models. It could also be used within a given model to determine which physical phenomena involved are most important to adequately model drug release.

6. Conclusion

The first CA model of a pharmaceutical polymeric device was introduced in 1990 by Zygourakis. Since then, several other models were developed. As of now, the release from 3D complex polymeric devices can be modelled using CAs while taking into account most critical independent variables such as porosity, solubility, polymer degradation, percolation, diffusion and spatial organisation.

However, the design of such models of drug release is very complex considering the large number of different independent variables. Even though the current models take most of these variables into account, it is not clear if the interdependence between the different variables has been adequately addressed. Therefore, these models certainly remain more descriptive than predictive. Furthermore, the treatment of the modeled phenomena is often a macro-scale empirical approximation of well-known molecular events.

Future trends in this field of research should focus on the modeling of fundamental molecular-scale events. The modeling of such events has been elegantly described by Kier in his different papers [11-13]. However, this approach was not scaled-up to describe polymeric drug delivery devices. Also, the development of these CA models requires very good computer programming skills and is far from being a simple task. There are certainly a lot of improvements that can be made to commoditize this technology and make it accessible to a larger number of pharmaceutical scientists. This could most likely be achieved by developing an easy to use yet flexible interface for the design of CA models of drug release from polymeric drug delivery devices. The implementation of such software would increase the scientific interest and accelerate the research in this field.

References

[1] Von Neumann, J. *Theory of Self-Reproducing Automata.* Urbana, London: University of Illinois Press, 1966.

[2] Gardner, M. Mathematical Games: The fantastic combinations of John Conway's new solitaire game "Life". *Scientific American.*, 1970, 223.

[3] Zygourakis, K. Development and temporal evolution of erosion fronts in bioerodible controlled release devices. *Chemical Engineering Science.*, 1990, 45, 2359-66.

[4] Zygourakis, K; Markenscoff, PA. Computer-aided design of bioerodible devices with optimal release characteristics: a cellular automata approach. *Biomaterials.*, 1996, 17, 125-35.

[5] Göpferich, A; Langer, R. Modeling polymer erosion. *Macromolecules.*, 1993, 26, 4105-12.

[6] Göpferich, A; Langer, R. Modeling of polymer erosion in three dimensions: Rotationally symmetric devices. *AIChE Journal.*, 1995, 41, 2292-99.

[7] Göpferich, A; Langer, R. Modeling monomer release from bioerodible polymers. *Journal of Controlled Release*., 1995, 33, 55-69.

[8] Göpferich, A. Polymer bulk erosion. *Macromolecules*., 1997, 30, 2598-604.

[9] von Burkersroda, F; Schedl, L; Göpferich, A. Why degradable polymers undergo surface erosion or bulk erosion. *Biomaterials*., 2002, 23, 4221-31.

[10] Göpferich, A. Erosion of composite polymer matrices. *Biomaterials*., 1997, 18, 397-403.

[11] Kier, LB; Cheng, CK; Testa, B. A cellular automata model for the percolation process. *Journal of Chemical Information and Computer Science*., 1999, 39, 326-32.

[12] Kier, LB; Cheng, CK; Testa, B; Carrupt, PA. A cellular automata model of micelle formation. *Pharmaceutical Research*., 1996, 13, 1419-22.

[13] Kier, LB; Cheng, CK; Testa, B; Carrupt, PA. A cellular automata model for diffusion in aqueous systems. *Journal of Pharmaceutical Sciences*., 1997, 86, 774-8.

[14] Barat, A; Ruskin, HJ; Crane, M. Probabilistic methods for drug dissolution. Part 2. Modeling a soluble binary drug delivery system dissolving in vitro. Simulation Modeling: *Practice and Theory*., 2006, 14, 857-73.

[15] Barat, A; Crane, M; Ruskin, HJ. Quantitative multi-agent models for simulating protein release from PLGA bioerodible nano- and microspheres. *Journal of Pharmaceutical and Biomedical Analysis*., 2008, 48, 361-8.

[16] Barat, A; Ruskin, HJ; Crane, M. 3D multi-agent modeles for protein release from PLGA spherical particles with complex inner morphologies. *Theory in Biosciences*., 2008.

[17] Vlugt-Wensik, KD; Vlugt, TJ; Jiskoot, W; Crommelin, DJ; Verrijk, R; Hennink, WE. Modeling the release of proteins from degrading crosslinked dextran microspheres using kinetic Monte Carlo simulations. *Journal of Controlled Release*., 2006, 111, 117-27.

[18] Bertrand, N; Leclair, G; Hildgen, P. Modeling drug release from bioerodible microspheres using a cellular automaton. *International Journal of Pharmaceutics*, 2007, 343, 197-207.

[19] Yu, R; Chen, H; Chen, T; Xiangyang, Z. Modeling and simulation of drug release from multi-layer biodegradable polymer microstructure in three dimensions. Simulation Modeling: *Practice and Theory*., 2008, 16, 15-25.

[20] Laaksonen, TJ; Laaksonen, HM; Hirvonen, JT; Murtomäki, L. Cellular automata model for drug release from binary matrix and reservoir polymeric devices. *Biomaterials*. 2009, 30, 1978-87.

[21] Laaksonen, HM; Hirvonen, JT; Laaksonen, TJ. Cellular automata model for swelling-controlled drug release. *International Journal of Pharmaceutics*, 2009, 380, 25-32.

[22] Göpferich, A. Mechanisms of polymer degradation and erosion. *Biomaterials*., 1996, 17, 103-14.

[23] Anderson, JM; Shive, M. Biodegradation and biocompatibility of PLA and PLGA microspheres. *Advanced Drug Delivery Reviews*., 1997, 28, 5-24.

[24] Ikada, Y; Tsuji, H. Biodegradable polyesters for medical and ecological applications. *Macromolecular Rapid Communications*., 2000, 21, 117-32.

[25] Shirahama, H; Ichimaru, A; Tsutsumi, C; Nakayama, Y; Yasuda, H. Characteristics of the biodegradability and physical properties of stereocomplexes between poly(L-

lactide) and poly(D-lactide) *Copolymers Journal of Polymer Science Part A*: *Polymer Chemistry*., 2005, 43, 438-54.

[26] Park, TG. Degradation of poly(D,L-Lactic acid) microspheres: effect of molecular weight. *Journal of Controlled Release*., 1994, 30, 161-73.

[27] Park, TG. Degradation of poly(lactic-*co*-glycolic acid) microspheres: effect of copolymer composition. *Biomaterials*., 1995, 16, 1123-30.

[28] Wu, L; Ding, J. In vitro degradation of three-dimensional porous poly(,-lactide-co-glycolide) scaffolds for tissue engineering. *Biomaterials*., 2004, 25, 5821-30.

[29] Siepmann, J; Faisant, N; Benoit, JP. A new mathematical model quantifying drug release from bioerodible microparticles using Monte Carlo simulations. *Pharmaceutical Research*., 2002, 19, 1885-93.

[30] Sandor, M; Enscore, D; Weston, P; Mathiowitz, E. Effect of protein molecular weight on release from micron-sized PLGA microspheres. *Journal of Controlled Release*., 2001, 76, 297-311.

[31] Yang, YY; Chung, TS; Ng, NP. Morphology, drug distribution, and in vitro release profiles of biodegradable polymeric microspheres containing protein fabricated by double-emulsion solvent extraction/evaporation method. *Biomaterials*., 2001, 22, 231-41.

[32] Tewes, F; Munnier, E; Antoon, B; Okassa, LN; Cohen-Jonathan, S; Marchais, H; et al. Comparative study of doxorubicin-loaded poly(lactide-co-glycolide) nanoparticles prepared by single and double emulsion methods. *European Journal of Pharmaceutics and Biopharmaceutics*., 2007, 66, 488-92.

[33] Aranaz, I; Gutierrez, MC; Yuste, L; Rojo, F; Ferrer, ML; del Monte, F. Controlled formation of the anhydrous polymorph of ciprofloxacin crystals embedded within chitosan scaffolds: study of the kinetic release dependence on crystal size. *Journal of Material Chemistry*, 2009, 19, 1576-82.

[34] Gaucher, G; Marchessault, RH; Leroux, JC. Polyester-based micelles and nanoparticles for the parenteral delivery of taxanes. *Journal of Controlled Release*., 2010, 142, 2-12.

[35] Klose, D; Siepmann, F; Elkharraz, K; Krenzlin, S; Siepmann, J. How porosity and size affect the drug release mechanisms from PLGA-based microparticles. *International Journal of Pharmaceutics,* 2006, 314, 198-206.

[36] Sant, S; Thommes, M; Hildgen, P. Microporous structure and drug release kinetics of polymeric nanoparticles. *Langmuir*., 2008, 24, 280-7.

[37] Sant, S; Nadeau, W; Hildgen, P. Effect of porosity on the release kinetics of propafenone-loaded PEG-g-PLA nanoparticles. *Journal of Controlled Release*., 2005, 107, 203-14.

[38] Fell, JT. Surface and interfacial phenomena. In: Aulton ME, editor. Autlon's Pharmaceutics: The design and manufacture of medicines. 3rd ed. Edinburgh: *Churchill Livingstone,* 2007, 59-69.

[39] Hurrell, S; Cameron, RE. Polyglycolide: degradation and drug release. Part I: Changes in morphology during degradation. *Journal of Material Science-Materials in Medicine*, 2001, 12, 811-6.

[40] Siepmann, J; Göpferich, A. Mathematical modeling of bioerodible, polymeric drug delivery systems. *Advanced Drug Delivery Reviews*, 2001, 48, 229-47.

[41] Siepmann, J; Elkharraz, K; Siepmann, F; Klose, D. How Autocatalysis Accelerates Drug Release from PLGA-Based Microparticles: A Quantitative Treatment. *Biomacromolecules*., 2005, 6, 2312-19.

[42] Siepmann, J; Siepmann, F. Mathematical modeling of drug delivery. *International Journal of Pharmaceutics*, 2008, 364, 328-43.

[43] Lao, LL; Venkatraman, SS; Peppas, NA. Modeling of drug release from biodegradable polymer blends. *European Journal of Pharmaceutics and Biopharmaceutics*. 2008, 70, 796-803.

[44] Berg, HC. *Random Walks in Biology*: Princeton University Press, 1983.

[45] Crank, J. *The Mathematics of Diffusion*. 2nd ed. Oxford: Oxford University Press; 1990.

[46] Kosmidis, K; Macheras, P. Monte Carlo simulations for the study of drug release from matrices with high and low diffusivity. *International Journal of Pharmaceutics*, 2007, 343, 166-72.

[47] Chopard, B; Droz, M; Kolb, M. Cellular automata approach to non-equilibrium diffusion and gradient percolation. *Journal of Physics A: Mathematical and General*., 1989, 22, 1609-19.

[48] Krishnan, K; Haddad, S; Pelekis, M. A simple index for representing the discrepancy between simulations of physiological pharmacokinetics models and experimental data. *Toxicology and Industrial Health*., 1995, 11, 413-21.

In: Cellular Automata
Editor: Thomas M. Li, pp.59-69

ISBN: 978-1-61761-592-4

Chapter 3

A MODEL OF CELLULAR AUTOMATA FOR THE SPATIAL ANALYSIS OF APHIDS AND LADYBUGS

Magda da Silva Peixoto[1*], Laécio Carvalho de Barros[2] and Rodney Carlos Bassanezi[3]

[1]Universidade Federal de São Carlos, Campus Sorocaba, Rodovia João Leme dos Santos, Sorocaba, SP, Brazil

[2]Departamento de Matemática Aplicada, IMECC, Universidade Estadual de Campinas, Campinas, SP, Brazil

[3]Centro de Matemática, Computação e Cognição, Universidade Federal do ABC Rua Santa Adélia, Santo André, SP, Brazil

ABSTRACT

In this chapter we have adopted the Cellular Automata model (CA) to study the dispersion of the aphids and ladybugs in the block of citric trees. The main aim of this investigation has been to develop a simple and specific methodology to study Citrus Sudden Death (CSD). CSD is a disease that has affected sweet orange trees grafted on *Rangpur lime* in the state of São Paulo - Brazil. Some studies suggest that this disease has been caused by a virus and it is transmitted by insects known as *aphids* (vector). The *ladybug* was selected among the most known enemies of aphids in citrus in Brazil. In order to elaborate a predator-prey type of model to study the interaction between aphids (preys) and ladybugs (predators) in citriculture we have used a fuzzy rule-based system (FRBS) [9]. Qualitative information from specialists permit to propose rules that relate the state variables with their own variations. The states of the variables of the system (inputs) are the density of preys and the potentiality of predators and their variations are the outputs. To study the temporal evolution of the disease we have adopted a (CA) model. Therefore, we take into account the effect of he wind in the space covered by the aphid, since the wind is important for the flight of the aphid [10]. Simulations were performed and compared between blocks with the presence of aphids and the absence of ladybugs, and the presence of both aphids and ladybugs. Numerical simulations allow us

[*] Corresponding author: magda@ufscar.br, laeciocb@ime.unicamp.br, rodney@ime.unicamp.br

to foresee the behavior of the system, hence creating a spectrum of possibilities and proposing control techniques for different initial scenarios.

After this introduction, the chapter proceeds as follows: Section 1 develops brief reviews of Citrus Sudden Death, Cellular Automata and the concept of fuzzy set and fuzzy rule-based system. Section 2 details cellular automata model. Simulations results are in Section 3.

1. PRELIMINARIES

1.1. Citrus Sudden Death

Brazil is currently the world greatest exporter of frozen and concentrated orange juice (74.6% of the production is exported). This leadership is directly connected to the quality and productivity of our orchards when compared to our competitors. Brazil is also the biggest world producer of orange juice with 49.3% of the total. The state of São Paulo is responsible for 97% of the Brazilian export and it is the biggest core of the Brazilian citric complex (of which 313 cities come from of the state of São Paulo and 7 come from Minas Gerais). São Paulo has the biggest citrus orchard complex in the world and it is responsible for more than 80% of the national production of the citric fruits. This agribusiness has been now facing a new threat: Citrus Sudden Death (CDS) [2].

In 1999 CSD was officially identified by the first time in the city of Comendador Gomes (state of Minas Gerais) and by the year of 2001 was detected by Fundecitrus (Fundo de Defesa da Citricultura – Araraquara/SP – Brazil) in the city of Colômbia (state of São Paulo). It means a threat for the national citriculture as long as it affects all commercial varieties of the sweet orange grafted on *Rangpur lime* which represent 85% of the orchard complexes of São Paulo and Minas Gerais [2].

CSD is a disease combining canopy/rootstock and it can lead sweet orange grafted on *Rangpur lime or Volkamer lemon* rootstocks to death [1]. Researches have shown that the ducts which lead nutrients generated by the photosynthesis to the roots become clogged and degenerated. The roots putrefy without nutrients and the tree decays and dies.

The initial signs of CSD are characterized by a generalized foliar discoloration. The trees also show partial defoliation, fewer new shoots and the absence of internal ones. These signs intensify as the disease develops and culminates with the death of the tree. The roots of the affected trees are characterized by the death of a large portion of the root system. The diagnostic confirmation of CSD has been done through the characteristic yellow stain in the phloem of the *Rangpur lime or Volkamer lemon* rootstocks [1].

According to [2] after the first signs of the disease, the number of symptomatic trees may reach between 60% and 100% of the block trees in less than 2 years. Normally, there is a rise of new symptomatic trees when the rainy season begins in the spring (from September to November) until the beginning of the summer (December and January). This number is reduced in the period between fall and winter (from March to August).

The CSD causing agent has not been confirmed yet. Researchers believe that insects known as aphids transmit the suspect virus responsible for CSD [12]. We have used the hypothesis that the aphids are the transmitters of the virus that causes CSD in this work.

The colonization of the orange trees is done by aphids that settle in the orchard. The aphids leave the block where they live when the block is saturated. The aphid flight in tropical weather is performed by winged aphids after several generations of insects without wings where are confined in the same habitat. Their flight is constant with an average of 6 to 12h or until their complete exhaustion for a day at most. The aphids follow horizontally the wind direction and lands on a new tree [3,13].

This flight depends on the local wind action [3]. Therefore the wind is a decisive factor for the insect dispersion and consequently for the disease once a test bite of an infected aphids is enough to transmit the virus to the tree.

The ladybug (predator) was selected among the most known enemies of aphids in citrus in Brazil. The main purpose of the study is to understand the dispersion of the aphids and ladybugs in the block of citric tree through the simulation model of Cellular Automata. Therefore, we intend to use the knowledge of the experts about the interaction between aphids and ladybugs, aiming for incorporating them sparingly into a simulation model with Cellular Automata.

1.2. Cellular Automata

Cellular Automata (CA) was introduced in the 1950s by the mathematician John von Neumann who considered the suggestions from Stanislaw Ulam and tried to model the natural process of self-reproduction [14].

Biological systems are ripe for modeling with CA methods. The spatial and temporal patterns are diverse and fascinating. Our knowledge of the details of a particular mechanism is often full of gaps and, unlike physics, there are few “laws” such as the Navier-Stokes equations or laws of Newton. Thus simplified models can be useful in precluding certain mechanisms as being impossible or at least unlikely. The speed by which calculations can be made allows the investigator to examine a huge number of parameter ranges that would be otherwise impractical for more “realistic” simulations [5]. In particular it can be employed in the ecological context to represent heterogeneous and dynamic habitat structures [4].

In short, CA consists of a simulation which is discrete in time, space and state of the systems. The idea of these models consists in considering each position (or region) of the spatial dominion as a cell which is attributed to a state. The state of each cell is modified regarding both its own state and the state of its neighbors during the former time stage through a series of simple rules that try to imitate the physical or the biological laws, which are, in our case, the ones that rules the system. In this approach the system state variables, as well as the time, are discrete.

We have used a Cellular Automata model to study the dispersion of the aphids and ladybugs in the block of citric tree. The system is represented in space through a cell lattice that interacts obeying some rules for the changing of the state among the cells. Each cell represents a tree that can be with the absence of aphids and ladybugs, the presence of aphids and the absence of ladybugs and the presence of aphids and ladybugs [11].

1.3. Fuzzy Rule-Based System

Fuzzy sets and fuzzy logic have become one of the emerging areas in contemporary technologies of information processing. Fuzzy Logic was first developed by [15] in the mid-1960s to represent uncertain and imprecise knowledge. It provides an approximate but effective means of describing the behavior of the system that is too complex, ill-defined, or not easily analyzed mathematically.

A fuzzy subset is characterized by a membership function mapping the elements of a domain, space, or universe of discourse *X* to the unit interval [0,1]. [15] That is, $\psi_A : X \to [0,1]$. Thus, a fuzzy set *A* in *X* may be represented as a set of ordered pairs of a generic element $x \in X$ and its grade of membership: $A = \{(\psi_A(x)/x)/x \in X\}$. Clearly, a fuzzy set is a generation of the concept of a set whose membership function takes on only two values {0,1}, that is, the characteristic function of A, $\chi_A : X \to \{0,1\}$.

Fuzzy variables are processed using a fuzzy rule-based system.

Fuzzy rule-based system (FRBS). Basically, fuzzy rule-based systems have four components: an input processor (fuzzification), a collection of linguistic rules called rule base; a fuzzy inference method and an output processor (defuzzification). These components process real-valued inputs in order to provide real-valued outputs. Figure 1 illustrates a FRBS.

The fuzzification is the process in which the input values of the system are translated into fuzzy sets of their respective universes. It is a mapping of the dominion of the real numbers led to the fuzzy dominion. Expert knowledge plays an important role to build the membership functions for each fuzzy set associated with the inputs.

The rule base characterizes the objectives and strategies used by specialists in the area through of a linguistic rule set. It is composed by a collection of fuzzy conditional propositions in the form if-then rules. An expert can articulate associations of linguist inputs/outputs.

The fuzzy inference method performs an approximate reasoning using the compositional rule of inference. A particular form of fuzzy inference of interest here is the Mamdani method [8]. In this case, it aggregates the rules through the logical operator OR, modeled by the maximum operator and, in each rule, the logical operators AND and THEN are modeled by the minimum operator [7]. The logic of decision to be made, incorporated to the structure of inference of the rule base, uses fuzzy implications [8] to simulate the decisions that are wanted. It generates actions inferred from consequents a set of input conditions – antecedents.

Finally, in defuzzification, the value of the output linguistic variable inferred from the fuzzy rule is translated to a real value. The output processor task is to provide real-valued outputs using defuzzification which is a process that chooses a real number that is representative of the inferred fuzzy set. A typical defuzzification scheme, the one adopted in this paper, is the centroid or center of mass method [11].

According to [4] fuzzy set approaches have been developed for special purposes where the information basis has been vague and imprecise. Under these conditions fuzzy techniques allow more accurate conclusions in comparison to the other approaches which cannot be applied successfully because of lack of data.

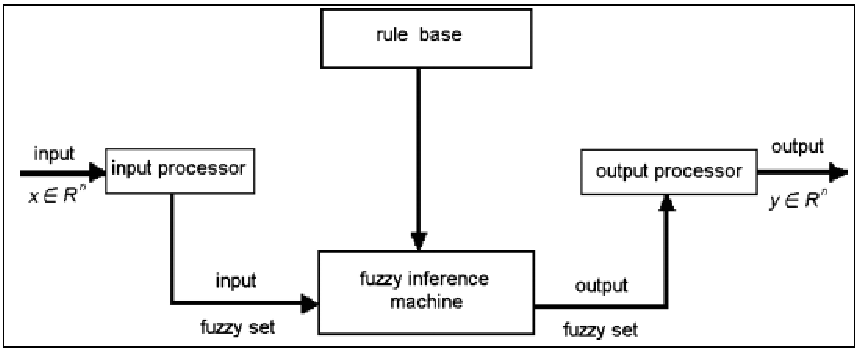


Figure 1. Structure of fuzzy rule-based system. Source: [6]

We suggest [7] and [8] for a detailed study of the fundamentals of fuzzy set and the systems theory and applications.

In the following section we have introduced the fuzzy model which is the main purpose of this chapter.

2. Cellular Automata Model

According to the observation made in section 2, the wind is a decisive factor for the insect dispersion and the information about the time of aphid flight is imprecise. In [10] we have adopted the CA model to study the temporal evolution of the disease. We take into account the effect of the wind in the space covered by the aphid. We have adopted the FRBS in order to establish the relationship between the space covered by the aphid and the intensity of the wind. In this work we have considered the hypothesis and the results of [10] to determine that the space covered by the aphid.

The attribute of each cell is described by two simultaneous values, one is the population density of aphids and the other is the potential of predation. Therefore, the attribute of dynamic system in the moment t is described by two matrixes. These matrixes are updated every single moment. Each cell of lattice represents a plant, that is, a tree.

The interaction between aphids and ladybugs occurs inside each cell through the predator-prey dynamic given by FRBS model of [9].

It is important to remember that the trees are shared uniformly by lines and columns in the block of citric trees.

Some hypotheses are essential to establish the rules of CA model:

- Brief contacts between the bodies of adults without wings facilitate the production of a greater number of winged descendents, in other words, it is supposed to come forth winged aphids when the population of these ones reaches its carrying capacity.
- The winged aphids create new colonies in other trees;
- Ladybugs are very inefficient in the capture of aphids as long as they only distinguish the presence of aphids when they are very close to their preys;

According to the previous hypotheses, we establish the further rules of movement:

1. Ladybugs capture aphids which are closer to them. The search is done among the "nearest neighbors". Each cell in two-dimensional square lattice has eight nearest neighbors conveniently addressed as points on the compass. The four neighbors sharing a common face are: N, E, S, and W. Those that can be reached diagonally are: NE, SE, NW, and SW. If they find aphids in one of these cells, the choice is random. Otherwise, the search is done until they find food (i.e. aphids).
2. Only adult ladybugs move among cells.
3. Winged aphids leave overcrowded cells and occupy other cells according to the dynamic of the flight established in [10], which is, according to the intensity of the wind.

We have used a Cellular Automata model to study the dispersion of the aphids and ladybugs in the block of citric tree. The system is represented in space through a cell lattice that interacts obeying the previous rules for the changing of the state among the cells. Each cell represents one tree that can be in one of the three states: presence of aphids and absence of ladybugs (gray), presence of aphids and ladybugs (black) and absence of aphids and ladybugs (white).

3. SIMULATIONS WITH CELLULAR AUTOMATA MODEL

In the numerical simulations the initial trees with aphids, ladybugs or both are chosen randomly. The initial number of insects in each tree is random as well. The block consists in 10x20 trees arranged homogenously.

The dispersion of the insects is plotted for a period of five days at each following figure.

Simulation 1: There are initially three trees with the presence of aphids and ladybugs (black) and seven trees with the presence of aphids and the absence of ladybugs (gray).

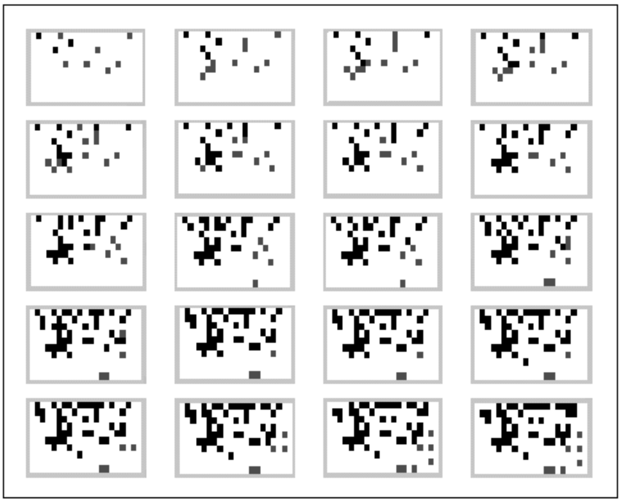


Figure 2. Presence of predator in the block. Initial condition: three trees with the presence of aphids and ladybugs and seven trees with the presence of aphids

Simulation 2: In this example we have accomplished two simulations. In the first one, there are aphids in one tree in the block and in the second, there are aphids and ladybugs.

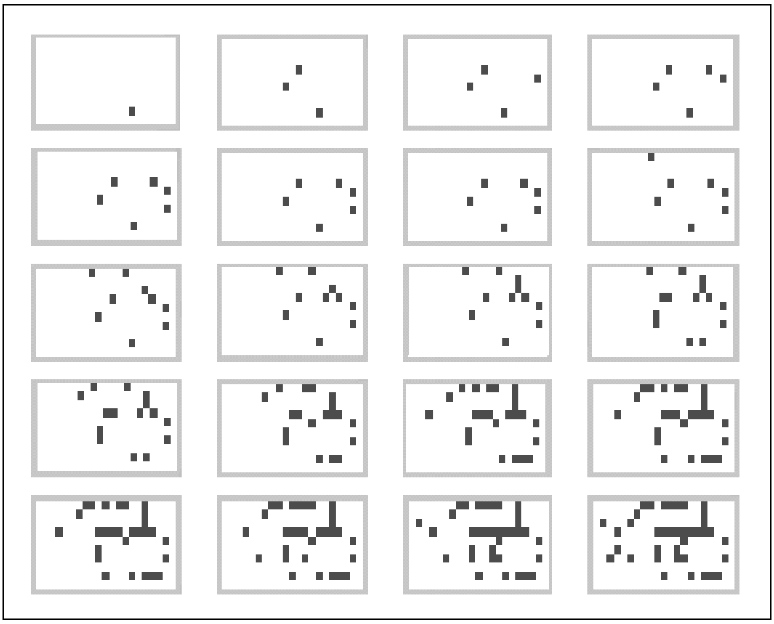


Figure 3. Absence of predator in the block. Initial condition: one tree with the presence of aphids

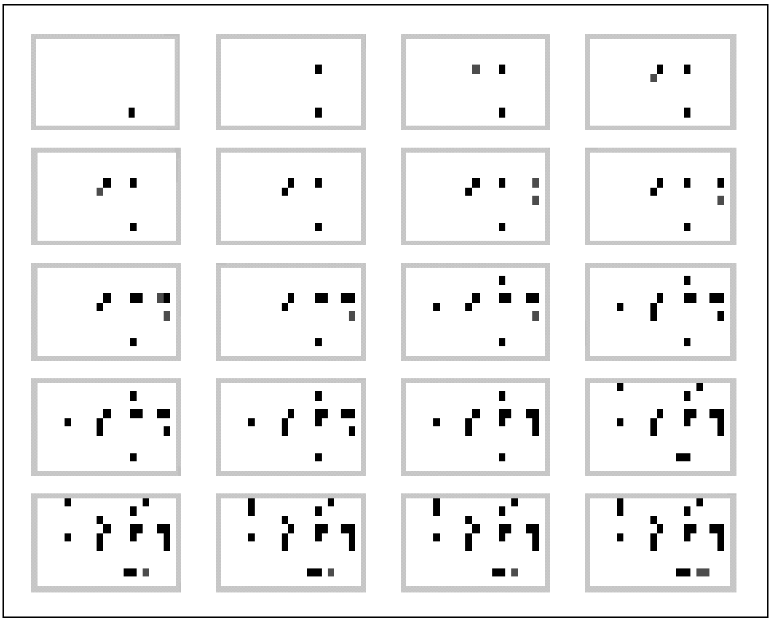


Figure 4. Presence of predator in the block. Initial condition: one tree with the presence of aphids and ladybugs

Simulation 3: Similar to the example 2, we have accomplished two simulations. There are aphids in five trees in the first one and in the second simulation, there are ladybugs in the same five trees in the block.

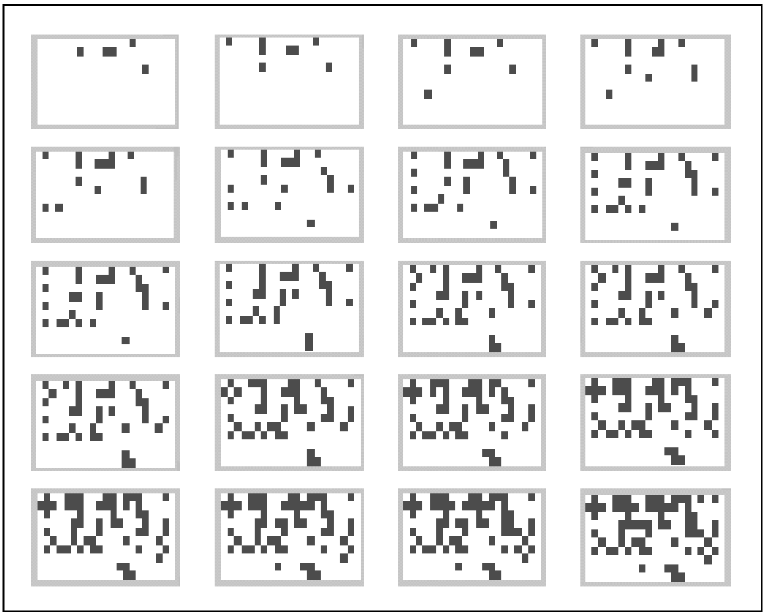


Figure 5. Absence of predator in the block. Initial condition: five trees with the presence of aphids

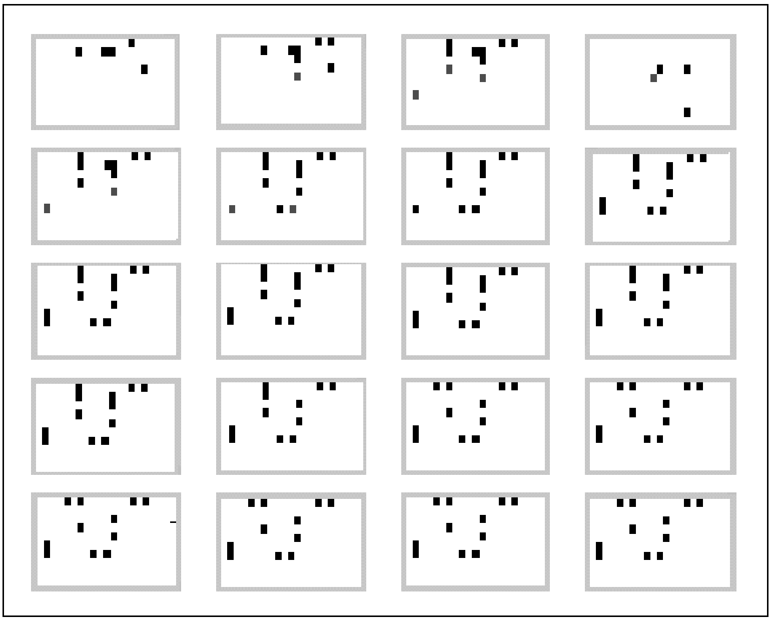


Figure 6. Presence of predator in the block. Initial condition: five trees with the presence of aphids and ladybugs

Simulation 4: This last example is similar to the example 3, but now there are ten trees under the conditions of the previous example.

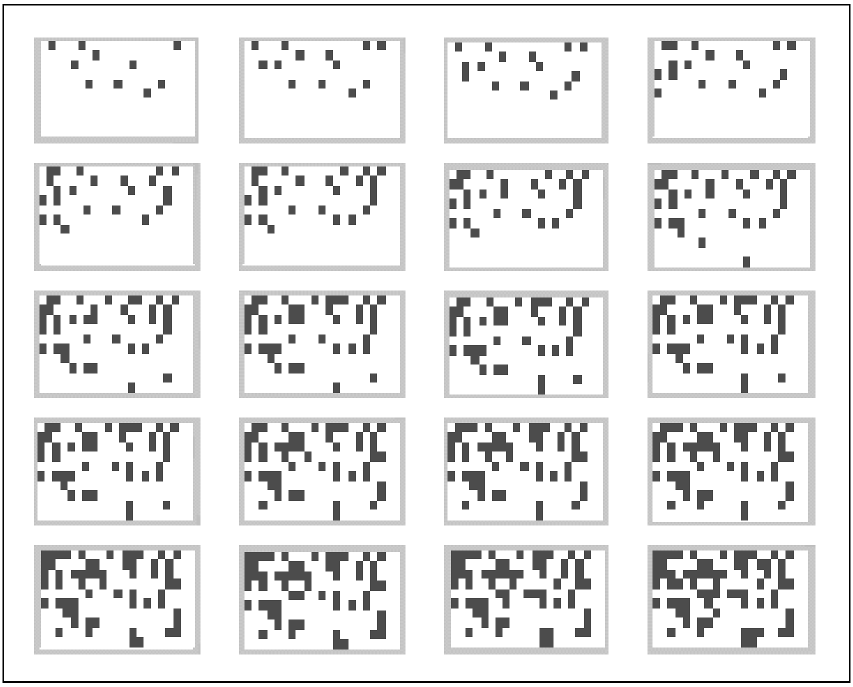


Figure 7. Absence of predator in the block. Initial condition: ten trees with the presence of aphids

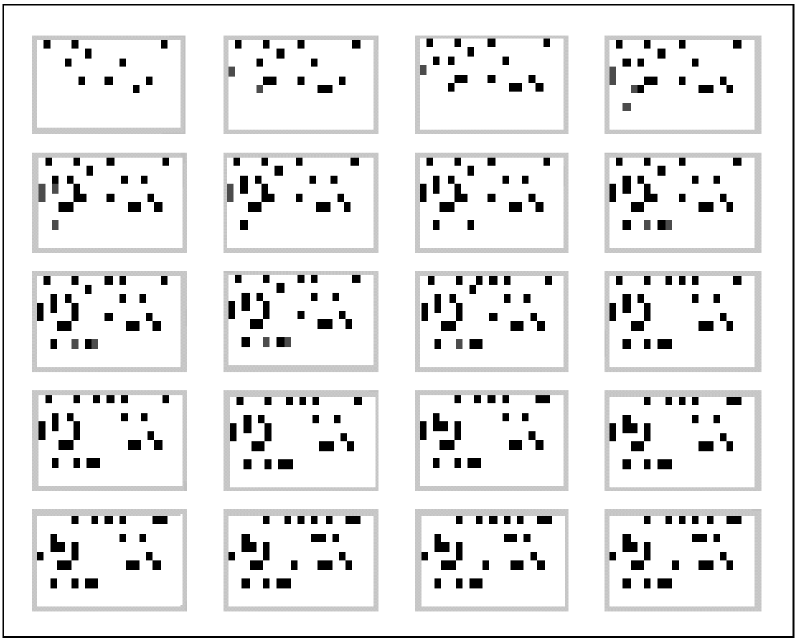


Figure 8. Presence of predator in the block. Initial condition: ten trees with the presence of aphids and ladybugs

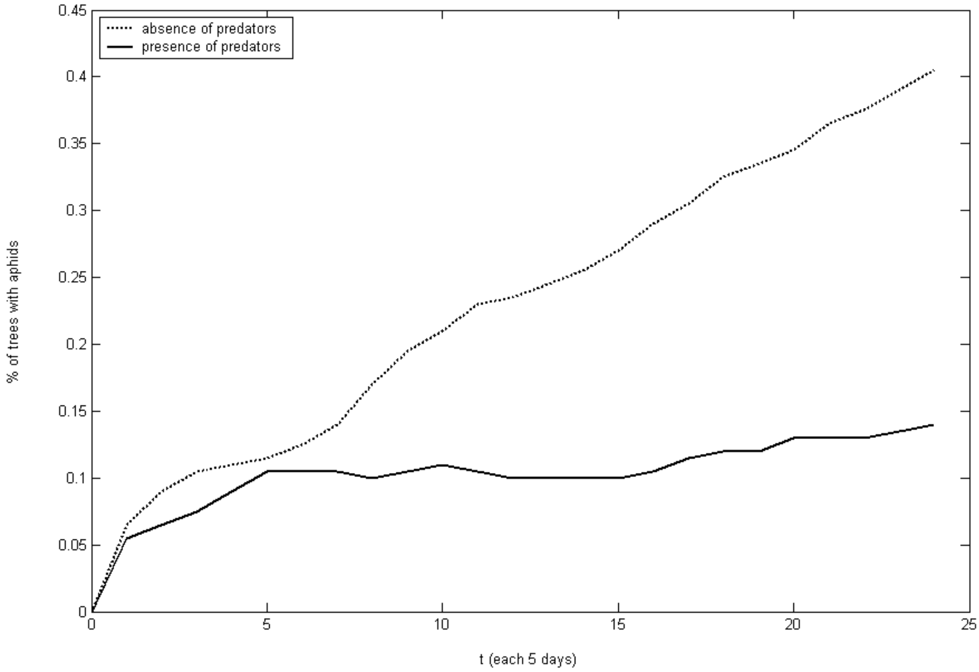


Figure 9. Percentage of trees with the presence of predators and the absence of predators

In the graphic (Figure 9) we have compared the difference between the quantity of tree with the presence of aphids and ladybugs and the trees with the presence of aphids and absence of ladybugs in the block.

Conclusions

We have evaluated the dynamics of aphids population with the presence of ladybugs (aphids predators). It is possible to observe this difference: the ladybugs are not capable of eliminating the entire population of aphids which is expected in a predator-prey model. It is considered impossible to eradicate the population of aphids, though it is probable to control its spread velocity. When we compare the simulations that have been done with Cellular Automata in each example and graph above.

Finally, the Cellular Automata approach developed in this work enhances understanding and contributes to investigate and to predict the spread of the insects which transmit the suspect virus responsible for Citrus Sudden Death.

Acknowledgments

The first author acknowledges Brazilian Research Council (CNPq) for the financial support.

REFERENCES

[1] Bassanezi, R. B., Gimenes-Fernandes, N., Yamamoto, P. T. & Jesus Jr., W. C. (2004). *Manual da Morte Súbita dos Citros*, Fundecitrus, São Paulo, Brazil (in Portuguese).

[2] Bassanezi, R. B., Bergamin Filho, A. & Amorim, L. (2003). Gimenes-Fernandes, N., and Gottwald, T.R., Spatial and Temporal Analysis of Citrus Sudden Death as a tool to generate hypothesis its etiology, *Phytopathology*, *93*, 502-512.

[3] Braga, A. & Sousa-Silva, C. R. (1999). *Afídeos de Citros (Citrus Sinensis) e seus predadores na região de São Carlos-SP*. Departamento de Ecologia e Biologia Evolutiva da Universidade Federal de São Carlos, São Carlos/SP, Brasil.

[4] Breckling, B., Muller, F., Reuter, H, Hölker, F. & Fränzle, O. (2005). Emergent properties in individual-based ecological models – introducing case studies in a ecosystem research context, *Ecol. Modell., 186*, 376-388.

[5] Ermentrout, G. B. & Edelstein-Keshet, L. (1993). Cellular Automata approaches to biological modeling, *Theor. Biol., 160*, 97-133.

[6] Jafelice, R. M., Barros, L. B., Bassanezi, R. C. & Gomide, F. (2003). Fuzzy modeling symptomatic HIV virus infected population, *Bull. Math. Biol.*, *66*, 1597-1620.

[7] Klir, G. J. (1995). *Yuan, B.-Fuzzy Sets And Fuzzy Logic: Theory and Applications.* Prentice Hall, N. Jersey.

[8] Pedrycs, W. & Gomide, F. (1998). *An Introduction to Fuzzy Sets: Analysis and Design,* Massachusets Institute of Technology.

[9] Peixoto, M. S., Barros, L. C. & Bassanezi, R. C. (2008). Predator Prey Fuzzy Model, *Ecological Modelling*, *v. 214*, 39-44.

[10] Peixoto, M. S., Barros, L. C. & Bassanezi, R. C. (2008). A Model Of Cellular Automata for the Spatial and Temporal Analysis of Citrus Sudden Death with the Fuzzy Parameter, *Ecological Modelling*, *v. 214*, 45-52.

[11] Peixoto, M. S. (2005). *Sistemas Dinâmicos e Controladores Fuzzy: Um Estudo da Dispersão da Morte Súbita dos Citros em São Paulo*, PhD Thesis, IMECC-Universidade Estadual de Campinas, Campinas/SP, Brazil, (in Portuguese).

[12] Revista Fapesp, 99, 2004 (in Portuguese), site accessed in May, 2004, http://www.revistapesquisa.fapesp.br.

[13] Tsai, J. H. & Wang, K. (1999). Life Table Study of Brown Citrus Aphids (Homoptera; Aphididae) at Different Temperatures, *Entomology Society of America*, pp. 412-419, 1999.

[14] Wolfram, S. (1994). *Cellular Automata and Complexity*, Addison-Wesley Publishing Company.

[15] Zadeh, L. A. (1965). *Fuzzy Sets. Informat. Control*, *8*, 338-353.

In: Cellular Automata
Editor: Thomas M. Li, pp.71-80

ISBN: 978-1-61761-592-4

Chapter 4

CELLULAR AUTOMATA OPTIMIZATION VIA EVOLUTIONARY METHODS

Epaminondas Sidiropoulos
Faculty of Engineering, Aristotle University of Thessaloniki, Greece

ABSTRACT

Cellular automata have been utilized for modeling and simulation of systems involving spatial arrangements. Examples include physical, social, urban and economic systems. The simulations are effected by means of local transition rules.

Cellular automata have also been employed as computation devices. For that purpose, genetic algorithms and simulated annealing were applied in order to determine suitable transition rules that would permit the cellular automaton to perform certain computational tasks. That approach constituted a form of optimization process.

However, in recent years optimization in a more general sense was presented in order to determine optimal configurations or arrangements of a cellular automaton. The latter may represent a physical or engineering system to be reformed or redesigned with a view to chosen objectives.

In the present approach, no specific transition rule is sought, but local transitions are guided by an evolutionary method suitably embedded into the automaton. Problems of spatial optimization have recently been treated by the author according to this methodology. In particular, location - allocation problems have been presented related to groundwater management, as well as land use management problems with or without allocation of resources.

The present communication proposes the application of an alternative evolutionary process based on mutating one by one all the components of the basic chromosome and selecting the best appearing result. This constitutes a local search to be combined with the special cell-based genetic algorithm described above. A general cellular – genetic framework is given for an algorithm suitable for spatial optimization problems. In an example problem, the superior performance of the proposed algorithm is shown.

INTRODUCTION

Cellular automata are two-dimensional constructs consisting of cells placed on a grid and endowed with a neighborhood structure. Each cell assumes a state, and a local transition rule determines the change of a cell's state from one discrete step to the next. It is important to emphasize that, according to the local rule, the transition from one state to another depends on the state of the cell in question and on the states of its neighboring cells. This kind of transition is applied to all cells in an evolution of the cellular automaton through successive time steps. These iterated local interactions yield interesting global results in an emergent fashion.

The special structure and the transition from the local to the global render the cellular automaton suitable for modeling and simulation to various physical as well as social and economic systems. Indeed, cellular automata have been used as the modeling framework for such diverse phenomena, both natural and anthropogenic, as forest fire propagation (Karafyllidis, 2004), urban development (Jennerette, G. D. and Wu, J., 2001), adsorption - diffusion processes (Chopard et al., 1989), fish migration (Schonfisch and Kinder, 2002), and soil erosion (D' Ambrosio et al. 2001).

In all those applications, the specification of suitable transition rules is the key issue. This fact is also true of another category of problems, which concern the design or regulation of a cellular automaton so that it optimizes the modeled system in a given sense. An early example of this idea is the derivation of transition rules such that the cellular automata can perform certain computational tasks (Mitchel et al., 1994).

Along the same line, topology optimization is an area of structural analysis in which, through cellular automata modeling, specific local transition rules are determined so as to achieve optimal arrangements. Finite element tessellations provide the cells and their neighborhoods. The transition rules are obtained through an optimization algorithm based on the finite element modeling (Abdalla, M. M. et al., 2006).

A more recent development is the application of the same paradigm in the water resources field. The operation of a reservoir system is modeled as a cellular automaton (Afshar and Shahidi, 2010). The objective of the optimization procedure is to regulate the water releases so as to satisfy as best as possible preset water demands. The problem is reduced to a local level thus allowing cellular local interaction and derivation of a suitable local transition rule.

According to a different, more general point of view, one does not seek to devise a specific rule or set of rules to apply for the evolution of the cellular automaton. Instead, the transitions of the cellular automaton will be guided through an evolutionary optimization process. The mode of transition will have a local character, as it suits cellular automata, but the transition operator will not be constant. It will be dictated each time by the evolutionary optimization process, until the overall combination of cell states satisfies certain preset goals.

In general terms, the optimization problem consists in determining the states of all cells in a cellular automaton, so as to achieve given objectives, possibly under given constraints. Land use planning problems fit naturally to this formulation: The area under study is divided into land blocks and it is desired to assign a land use to each block, so as to satisfy economical and ecological goals. The problem of allocating resources among the land blocks may also be treated under the same framework.

A problem of this category in the context of groundwater allocation and simultaneous water well location was presented by Sidiropoulos and Tolikas (2008). In their approach the authors embedded a genetic algorithm into the cellular automaton, so that the genetic algorithm guided the evolution of the cellular automaton through successive iterative steps. A related problem was presented by Sidiropoulos and Fotakis (2009). It concerned groundwater allocation among land blocks arranged in the form of a cellular automaton. Pumping and transportation cost was to be minimized. A specially designed genetic algorithm was again embedded into the cellular automaton. This genetic algorithm preserved local features of the automaton. A more conventional genetic algorithm was also employed that did not include local features but it was shown to be less efficient.

In this communication the above special algorithm is complemented by a local search procedure yielding better optimization results. The whole scheme is discussed and presented under a more general framework.

CELLULAR FORMULATION

A cellular automaton is formally represented as a discrete set of sites, each one of which is occupied by a cell:

$$\mathbf{L} = \left\{(i,j) \,|\, 1 \le i \le I, 1 \le j \le J\right\} \tag{1}$$

For each cell a neighborhood is defined as:

$$\mathbf{N}(i,j) = \left\{(i+\Delta i, j+\Delta j) \in \mathbf{L} \,|\, \Delta i \in \{-1,0,1\}, \Delta j \in \{-1,0,1\}\right\} \tag{2}$$

The above represents the Moore neighborhood including eight neighboring sites and the central site itself. The von Neuman type neighborhood consisting of the four North – South and East – West sites and the central site itself can be represented as:

$$\begin{aligned} &\mathbf{N}(i,j) = \{(i+\Delta i, j+\Delta j) \in \mathbf{L} \,|\, \Delta i \in \{-1,0,1\}, \Delta j \in \{-1,0,1\} \\ &\text{and} \quad \Delta i \cdot \Delta j = (1-|\Delta i|)(1-|\Delta j|)\} \end{aligned} \tag{3}$$

Let $\sigma^{(t)}(i,j) \in S$ be the state of cell (i,j) at time step t, where S is the set of possible states, typically a discrete set. Also, let

$$\sigma^{(t)}(\mathbf{N}(i,j)) = \{\sigma^{(t)}(k,l) \,|\, (k,l) \in \mathbf{N}(i,j)\} \tag{4}$$

be the set of states assumed by the cells of the neighborhood of (i,j) at time step t.

Then the transition of the cell (i,j) to time step t+1 may be denoted as

$$\sigma^{(t+1)}(i,j) = \tau(\sigma^{(t)}(\mathbf{N}(i,j)) \tag{5}$$

where τ is the transition function that depends only on the states of the neighboring cells. The function τ is applied to each one of the cells of the automaton and in each case the states at time t enter the argument of the function τ. This mode of transition is characterized as synchronous updating. Let

$$\mathbf{K}^{(t)} = \{\sigma^{(t)}(i,j) \,|\, (i,j) \in \mathbf{L}\} \tag{6}$$

be the set of states of all cells at time t. Then the application of τ to the cells of the automaton via Equation (5) will lead $\mathbf{K}^{(t)}$ of Equation (6) to $\mathbf{K}^{(t+1)}$. Therefore, the overall effect of the local transition rule (5) may be considered as the application of an operator **O** to $\mathbf{K}^{(t)}$:

$$\mathbf{K}^{(t+1)} = \mathbf{O}[\mathbf{K}^{(t)}] \tag{7}$$

The category of problems to be treated involves a cellular automaton representation and an optimization of the cell state distribution. For a genetic algorithm solution, the application of genetic operators will result in modifications of the cellular mosaic, eventually leading to optimal distributions or configurations. In the next section the operator of Equation (7) will be made more specific for the case of transitions via genetic algorithms.

COMBINED CELLULAR – GENETIC FORMULATION

The typical problem of spatial optimization involves an array of land blocks. The distribution of land uses among these blocks is sought, such that it satisfies certain preset goals. The objective function of such a problem depends on the land uses to be assigned to the blocks. A problem of this type involving resource allocation among land blocks has been presented by Sidiropoulos and Fotakis (2009).

The land blocks in this type of a problem may be considered as cells with a well defined neighborhood structure and the land use or any other relevant characteristic of the block may be identified with the state of that block. Based on this consideration a combined genetic – cellular formulation of the problem and of an algorithm for its solution is given below.

Let (x_i,y_i) be the coordinates of the centers of the land blocks, where $1 \le i \le I$ and $1 \le j \le I$. Then the blocks may be numbered consecutively using the index

$$k = (j-1)I + i \tag{8}$$

Obviously, by taking one row after the other, $k = 1, 2, ..., I \cdot J$

The same index can now be applied to the state of the cell (i,j), as well as to the corresponding neighborhood, as the latter was defined by Equation (2):

$$\sigma(k) = \sigma(i,j)\,,\ \mathbf{N}(k) = \mathbf{N}(i,j)$$

where k is given in terms of i and j via Equation (8).

The state of the whole cellular automaton may now be represented in the form of a one - dimensional array:

$$\mathbf{C} = \{\sigma(k) \mid k = 1,2,...,I \cdot J, \sigma(k) \in \{1,2,..,m\}\} \tag{9}$$

The objective function F(.) of the spatial optimization problem depends on the distribution of the states, i.e. in terms of Equation (9),

$$F = F(\mathbf{C}) \tag{10}$$

A natural genetic algorithm for the solution of the problem would have the representation (9) as its typical chromosome, i.e.

$$\mathbf{C}_p = \{\sigma_p(k) \mid k = 1,2,...,I \cdot J\} \tag{11}$$

where the index p runs through the population of the chromosomes: p=1,2,..,P with P being the size of the population.

This kind of algorithm was used by Strange et al. (2001) in a land use optimization treatment under cellular formulation and, also, by Seppelt and Voinov (2003) in the context of ecosystem optimization. The encoding of the type (11), although cellular in nature, does not entail the local aspect of the cellular automaton. The present approach is based on an alternative chromosome structure, namely,

$$\mathbf{D}_p = \{n_k \mid n_k \in \mathbf{N}(k),\ \ k = 1,2,...,I \cdot J\}\text{, p=1,2,..,P.} \tag{12}$$

In the latter representation, for each cell c_k a neighboring cell is selected. Eventually, the state of this neighboring cell will be adopted by the cell in question.

In order to evaluate the chromosomes of the above type (12), $\mathbf{D}_p$ is considered as an operator acting on a base mosaic of the type (9). Indeed, by taking

$$\sigma'(k) = \sigma(n_k)$$

A new array is formed

$$\mathbf{C}'_p = \{\sigma'(k) \mid k = 1,2,...,I \cdot J\}\text{, p=1,2,..,P.} \tag{13}$$

Thus, the effect of $\mathbf{D}_p$ can be written as an operator acting on $\mathbf{C}$:

$$\mathbf{C}'_p = \mathbf{C} \otimes \mathbf{D}_p \tag{14}$$

Then, by applying Equation (10), the values of the chromosomes (12) would become equal to

$$v_p = F(\mathbf{C}'_p)\,,\ p=1,2,..,P \tag{15}$$

If the problem is one of maximization, then the best chromosome is the one with the largest value. Let pbest be such that

$$v_{\text{pbest}} = \max_{1\le p\le P}\{F(\mathbf{C}'_p)\} \tag{16}$$

Then $\mathbf{C}'_{\text{pbest}}$ is the best cellular arrangement induced by the population of chromosomes (12).

Consequently, the configuration $\mathbf{C}'_{\text{pbest}}$ is taken as the new base configuration. This is precisely the result of the major algorithmic step that takes one configuration to the next. Each such step includes a generation of the genetic algorithm. This effect can be expressed as follows, if the successive discrete steps are indexed by a superscript t:

$$\mathbf{C}^{(t+1)} = \mathbf{O}[\mathbf{C}^{(t)}] \tag{17}$$

Equation (17), in operator form, summarizes the effect of Equations (11) through (16) and identifies in specific terms the general operator formulation of the previous section (Equation 7).

Equation (15) expresses the evaluation of the chromosomes, necessary for the genetic operator of selection. Subsequently, the genetic operators of crossover and mutation are applied to the population of chromosomes of the type (12) and a new population of chromosomes results. These new chromosomes are of the form (12) and are applied to the new base configuration in the sense of Equation (14) and so on. It is noted again that the above discrete time step t coincides with a generation of the genetic algorithm.

The above procedure can be summarized by through the particular steps of the following cell- based genetic algorithm (CBGA):

(a) An initial mosaic of the type (9) is formed by means of random number generation. It becomes the base configuration.
(b) A random initial population of chromosomes of the type (12) is formed.
(c) A number of P new mosaics are formed by applying each one of the chromosomes to the base configuration in the sense of (14).
(d) The best of the mosaics of (c) becomes the new base configuration.
(e) Stopping criteria are applied and if they are satisfied the algorithm is terminated.
(f) The chromosomes of the population are subjected to the standard genetic operators of selection, crossover and mutation.
(g) A new population of chromosomes results and the algorithm is directed to (c).

A stopping criterion is formulated as follows: While running the algorithm from generation to generation, a record is kept of the best five base configurations obtained up to the current generation. The record is updated each time just after step (d) above. In the updated record the degree of similarity is computed between the best and the worst member of the record. The degree of similarity between two mosaics is defined as the percentage of homologous cells having the same state. If the degree of similarity exceeds a given level or if the maximum number of generations has been reached, the algorithm is terminated.

A modified version of CBGA is obtained if the new base configuration of step (d) is not the best one produced by the current population of chromosomes, but the best one achieved up to the current generation. The latter is to be found in the updated record of the best configurations.

LOCAL SEARCH ALGORITHM

It is well known that genetic algorithms are good at reaching the area of the optimum quite rapidly but are rather slow in determining the position of the optimum more precisely. For this reason genetic algorithms are very often supplemented by local search algorithms. The latter perform neighborhood searches in an effort to achieve improvement and local tuning of search results. Hart et al. (2005) give both the principles and the techniques of such algorithms pointing out that, starting from an initial solution, solutions neighboring to a given one are generated until a better point in the solution space is found.

In a problem of spatial optimization, neighboring solutions can be generated by modifying the states in one or more cells of the cellular automaton that models the spatial field. In contrast to the genetic algorithm presented in the previous section, it was found more advantageous in the local search to perform updating in an asynchronous fashion. More specifically, the following local search algorithm is proposed:

The algorithm starts with an initial base configuration of the type (9). For each each cell k, its neighborhood **N**(k) is considered and all states of the neighborhood cells are modified randomly and asynchronously. The resulting mosaics are evaluated and the best one is singled out. Finally the overall best one is found and if it is better than the initial one, it takes its place as the base configuration.

In pseudocode form:

Let l_b and f_b be the base configuration and its value, respectively.

Let $l(0) = l_b$ and $f(0) = f_b$

```
For k=1 to I·J
    l(k)=l(k-1) and f(k)=f(k-1)
    Let $s_k$ be the size of the neighborhood N(k)
    For j=1 to $s_k$
        Let kj be the jth site of N(k)
        Let $r_{kj}$ be a random integer between 1 and m
        Replace the state at kj with $r_{kj}$
        Let $l_{kj}$ be the corresponding modified chromosome
        Let fk(j)=F($l_{kj}$) be the value of $l_{kj}$
    End For
    Let $f_{kh} = \max_{1 \le j \le s_k} \{fk(j)\}$
    If $f_{kh}$>f(k) then f(k)= $f_{kh}$
    End For
    Let $f_h = \max_{1 \le k \le I \cdot J} \{f(k)\}$
    If $f_h > f_b$ then $l_b = l_h$ and $f_b = f_h$.
```

The process is repeated until a stopping criterion is satisfied.

In the present approach the local search algorithm was used to improve the results of the genetic algorithm.

RESULTS AND DISCUSSION

A specific example problem of spatial resource allocation is presented here. A 10x10 grid includes 100 blocks that receive water from 3 wells. The wells are placed in the following positions:

$$x_{w1}=17,\ y_{w1}=5, \quad x_{w2}=18,\ y_{w2}=0, \quad x_{w3}=15,\ y_{w3}=0$$

The hydraulic conductivities in the areas around the wells are $k_1=0.7\text{x}10^{-3}$m/s, $k_2=0.5\text{x}10^{-3}$m/s, $k_3=1.2\text{x}10^{-3}$m/s. The thickness of the aquifer was taken equal to b=50 m, the radius of influence R = 15 m and the radii of the wells all equal to r_w = 0.10 m.

The objective of the problem is to assign one wells to each one of the blocks, so that the cost of pumping and transportation will be minimized. The objective function of the problem has been given in detail by Sidiropoulos and Fotakis (2009). A hydrodynamic model of the underlying aquifer has also been given, showing that the effects of the wells on the objective function are interdependent and, therefore, non-separable.

For the solution of the problem three algorithms were tried: (1) The natural genetic algorithm characterized by the chromosome of Equation (11), (2) the CBGA and (3) the modified CBGA, supplemented by the local search algorithm. Figures 1, 2 and 3 show the respective resulting configurations. The configuration of Figure 1 was reached after 500 generations and the one of Figure 2 after 700 generations. The configuration of Figure 3 was obtained in two stages. First the modified CBGA took 300 generations to reach a similarity degree of 98% and then the local search algorithm was employed using the best result of the genetic algorithm as a starting configuration. The depicted configuration was reached after 22 steps of the local search algorithm. The best value of the objective function was obtained from the combined application of modified CBGA and local search. The natural genetic algorithm lags behind in efficiency and in quality of final result in comparison to CBGA. Moreover, it is observed that the mosaic shown in Figure 3 presents not only compact, but also contiguous areas.

It is noted here that the present approach does not involve any assumptions on weak interactions among land blocks (Seppelt and Voinov, 2003) and, also, no objective functions of local nature are used. The typical chromosome of CBGA introduces a kind of local structure, but the objective function applies to the totality of the cellular automaton.

The issue of obtaining emergent results from the local to the global level has been addressed by Fotakis (2009) and Fotakis and Sidiropoulos (2009). Questions of multi-objective spatial optimization have been treated by Fotakis (2009).

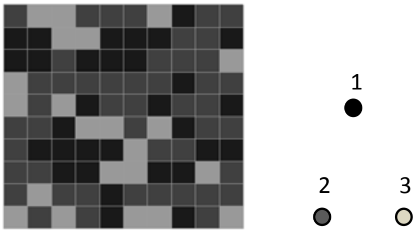


Figure 1. Natural genetic algorithm

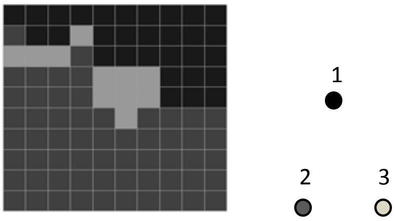


Figure 2. CBGA

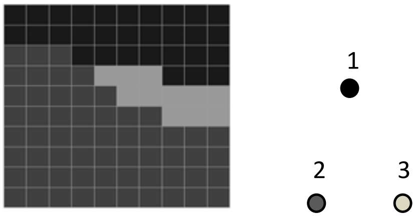


Figure 3. modified CBGA with local search

In conclusion, there are many more spatial optimization problems that call for extensive applications of the algorithms presented in this short communication.

REFERENCES

[1] Abdalla, M. M., Setoodeh, S. & Gürdal, Z. (2006). Cellular automata paradigm for Topology optimization. IUTAM Symposium on Topological Design Optimization of Structures, *Machines and Materials: Status and Perspectives*, 169-180.

[2] Chopard, B., Droz, M. & Kolb, M. (1989). Cellular automata approach to non-equilibrium diffusion and gradient percolation, *Journal of Physics, A: Mathematical and General*, *22*, 1609-1619.

[3] D' Ambrosio, D., Di Gregorio, S., Gabriele, S. & Gaudio, R. A cellular automata model for soil erosion by water, *Physics and Chemistry of the Earth* (B), *vol 26*, No 1, 33-39.

[4] Fotakis, D. (2009). "Spatial planning for decision making. Combined management of land use and water resources via evolutionary methods" (in Greek), Doctoral dissertation, Faculty of Engineering, Aristotle University, Thessaloniki.

[5] Fotakis, D. & Sidiropoulos, E. (2009). *Combined land-use and water allocation planning.*, submitted for publication.

[6] Hart, W. E., Krasnogor, N. & Smith, J. E. Editors, (2005). *Recent Advances in Memetic Algorithms*, Springer Studies in Fuzziness and Soft Computing.

[7] Jennerette, G. D. & Wu, J. (2001). Analysis and simulation of land use change in the central Arizona – Phoenix region, USA. *Landscape Ecology*, *16*, 611-626.

[8] Karafyllidis, I. (2004). Design of a dedicated parallel processor for the prediction of forest fire spreading using cellular automata and genetic algorithms. *Engineering Applications of Artificial Intelligence*, 17, 19-36.

[9] Mitchel, M., Crutchfield, J. & Hraber, P. T. (1994). Evolving cellular automata to perform computations, *Physica, D75*, 361-391.

[10] Schonfisch, B. & Kinder, M. (2002). A fish migration model. *Proceedings of the 5th International Conference on Cellular Automata for Research and Industry,* 9-11 October, Springer.

[11] Seppelt, R. & Voinov, A. (2003). Landscape optimization: Application of a spatial ecosystem model, Chapter 12 in Costanza, R. and Voinov, A., Editors, *Landscape Simulation Modeling*, Springer.

[12] Sidiropoulos, E. & Tolikas, P. (2008). Genetic algorithms and cellular automata in aquifer management, *Applied Mathematical Modelling*, *32*, 617-640.5

[13] Sidiropoulos, E. & Fotakis, D. (2009). Cell-based genetic algorithm and simulated annealing for spatial groundwater allocation. WSEAS *Transactions on Environment and Development*, *vol.5*, issue 4, 351-360.

[14] Strange, N., Meilby, H. & Bogetoft, P. (2001). Land use optimization using self-organizing algorithms. *Natural Resource Modeling*, *vol. 14*, no 4.

In: Cellular Automata
Editor: Thomas M. Li, pp.81-90

ISBN: 978-1-61761-592-4

Chapter 5

PARALLEL CELLULAR AUTOMATA ON CHIP

Juan A. Gomez-Pulido*[*], *Miguel A. Vega-Rodríguez and Juan M. Sánchez-Pérez

Department of Technologies of Computers and Communication,
University of Extremadura, Caceres, Spain

ABSTRACT

Cellular Automata are often used as methodology to approach complex optimization problems in the Artificial Intelligence area, where some parts of the problem can be performed in a parallel way. In many cases, the complex nature of the problem can require high computational effort, so techniques that increase the performance are welcome. A possible solution that would allow us to accelerate the computation is to provide of a custom hardware processor designed for implementing several automaton running in parallel. To evaluate this alternative, we have designed and verified a reconfigurable processor that runs a stand-alone Cellular Automaton; hence many of these can work in parallel on the same chip. In order to obtain a valid prototype of the reconfigurable processor, we have considered the Conway's Game of Life as example of a simple Cellular Automaton.

Keywords: Cellular Automata; Conway's Game of Life; Parallelism; Reconfigurable Computing.

1. INTRODUCTION

Cellular Automata (CA) are computational systems often used as methodology satisfactorily applied for solving combinatorial optimization problems in the Artificial Intelligence area [1][2][3]. However, a CA used for solving a large problem may have a very high computational cost, even when it is run on a high-performance machine. Taking into

[*] Corresponding author: jangomez@unex.es

account that many optimization problems could be tackled by parallel methodologies [3], we have developed a specific purpose processor that runs a CA in a stand-alone way in order to implement a set of CA working in parallel in the same chip, using the reconfigurable hardware technology. This way the computer, besides allowing the monitoring and control of the CA-based system, can be used for any another task with the whole potential of their resources. The combination of parallelism and hardware implementation allows increasing the speed of the system regarding the one implemented by software and performed on a general purpose computer. Moreover, the possibility to release to the host of the effort in running the algorithm is also a sufficiently interesting thing as to attempt this alternative.

In this work we describe the hardware implementation, using reconfigurable devices, of a simple CA for solving the Conway's Game of Life, because this simple case can show us the ability of the used tools to model this kind of systems. This simple CA is just used as basis of a multi-CA system on chip.

The rest of the paper is organized as follows: Section 2 describes briefly the "Game of Life" characteristics. In Section 3, we introduce the reconfigurable computing technology. Then, Section 4 shows the tools and techniques used for the development of the reconfigurable processor. Finally, Section 5 presents the experimental results, while Section 6 exposes the conclusions and the future work.

2. A Simple Cellular Automaton

We use The Game of Life (also named Life) as example of simple CA to be implemented by means of a specific-purpose hardware processor. Life was formulated by the British mathematician John Horton Conway in 1970 [4]. From then on it constitutes an example of CA very divulged in academic environments.

Under the user point of view, Life shows a pattern easily evolving in pretty forms, but for the researcher the advantage of Life consists of its meaning as a basic example of self-organization principle from some few rules. We should not forget the interesting ability of Life to compute any algorithm, which can be used, for example, to approach certain combinatorial optimization problems.

The behavior of Life is simple. From starting positions the patterns evolve indefinitely, or until reaching a stop condition. In some cases the evolution could imply a very high number of iterations, maintaining busy the computer during days. Life does not need any direct instruction from the user, since the pattern evolution is determined by the initial state and for the rules of the game. It runs on a grid of square cells which enlarges to infinity in all directions in discrete times that are the iterations of the algorithm. Each cell has eight neighbors (the adjacent cells) and can be in one of two states: it is either live (state on) or dead (state off). When iteration ends, all the cells are updated in parallel at the same time (the on-off transitions occur simultaneously).

We use two possible cases of CA to implement the hardware processor, regarding to the fixed size of the grid of square cells: 20x20 or 30x30. The variable DIM takes the value 20 or 40 for these cases. For any of these cases, we have an initial pattern of 12 live cells forming a cross around the center of the grid. The evolution of the first iterations of an example of simple Life can be observed in Figure 1.

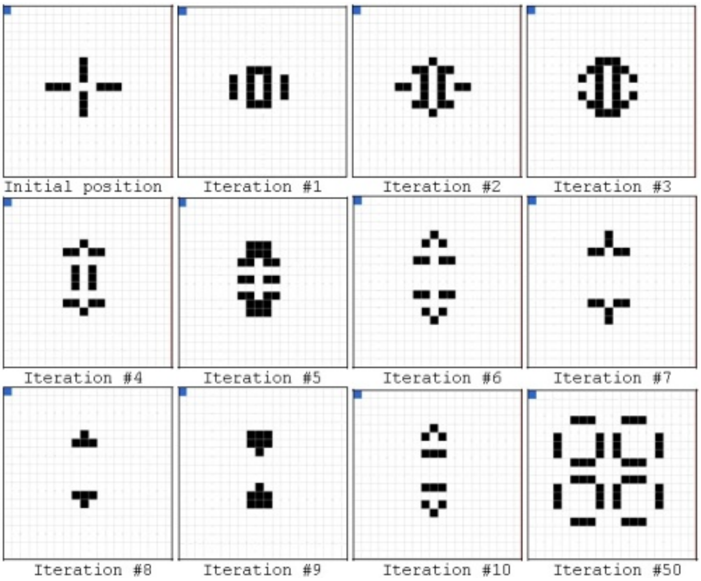


Figure 1. Evolution of the first iterations of the Game of Life

3. RECONFIGURABLE COMPUTING

Reconfiguration of circuitry at runtime to suit the application at hand has created a promising paradigm of computing that blurs traditional frontiers between software and hardware. This powerful computing paradigm, named reconfigurable computing, is based on the use of programmable logic devices, mainly field programmable gate arrays (FPGAs) [5] incorporated in board-level systems. FPGAs have the benefits of the hardware speed and the software flexibility; also, they have a price/performance ratio much more favourable than ASICs (Application-Specific Integrated Circuits). For these reasons, FPGAs are a good alternative for many real applications in image and signal processing, multimedia, robotics, telecommunications, cryptography, networking and computation in general [6].

Furthermore, as the reconfigurable computing is becoming an increasingly important computing paradigm, more and more tools are appearing in order to facilitate the FPGA programmability using higher-level HDLs (Hardware Description Languages). In this line, several research projects have been developed to bring new high-level languages, most notably, SpecC [7], SystemC [8] or Ocapi-xl [9]. And on the other hand, several companies are proposing their own high-level HDLs, such as Handel-C [10]. The main advantage of all these new hardware description languages is their simplicity, where the hardware design can be defined and evaluated using a pseudo-C programming style. In this way, FPGA devices are making it possible for thousands of computer engineers to have access to digital design technology in an easier way, obtaining a better performance with a similar flexibility to software. In addition, ASIC engineers are now ‘reconfiguring' themselves as FPGA engineers for economic reasons and adding to the growing legions of FPGA designers around the world.

In our research works we use real hardware resources. The details of the used prototyping boards are shown in Table 1.

Table 1. Hardware resources used in this work, arranged by reconfigurable versus general-purpose hardware with similar technology in order to do an effective comparison of results

Technology		Reconfigurable computing		General purpose processor	
CMOS	Year	FPGA device [www.xilinx.com]	Board	Processor	Computer
130 nm	2002	Xilinx Virtex2 Pro - xc2vp30-7ff896	Digilent XUPV2P [www.digilentinc.com]	Intel Pentium4 2.4GHz	1 GB RAM, Win2003
65 nm	2006	Xilinx Virtex5 LT330 - xc5vlx330-1ff1760	PLDA PCIXSYSV5 [www.plda.com]	Intel Core2 Duo E4500 2.2GHz	2 GB RAM, WinXP

4. Cellular Automata Reconfigurable Processor

The followed process for the development of a processor based on reconfigurable technology that implements Life implies some stages that we explain next, and that they are shown in Figure 2.

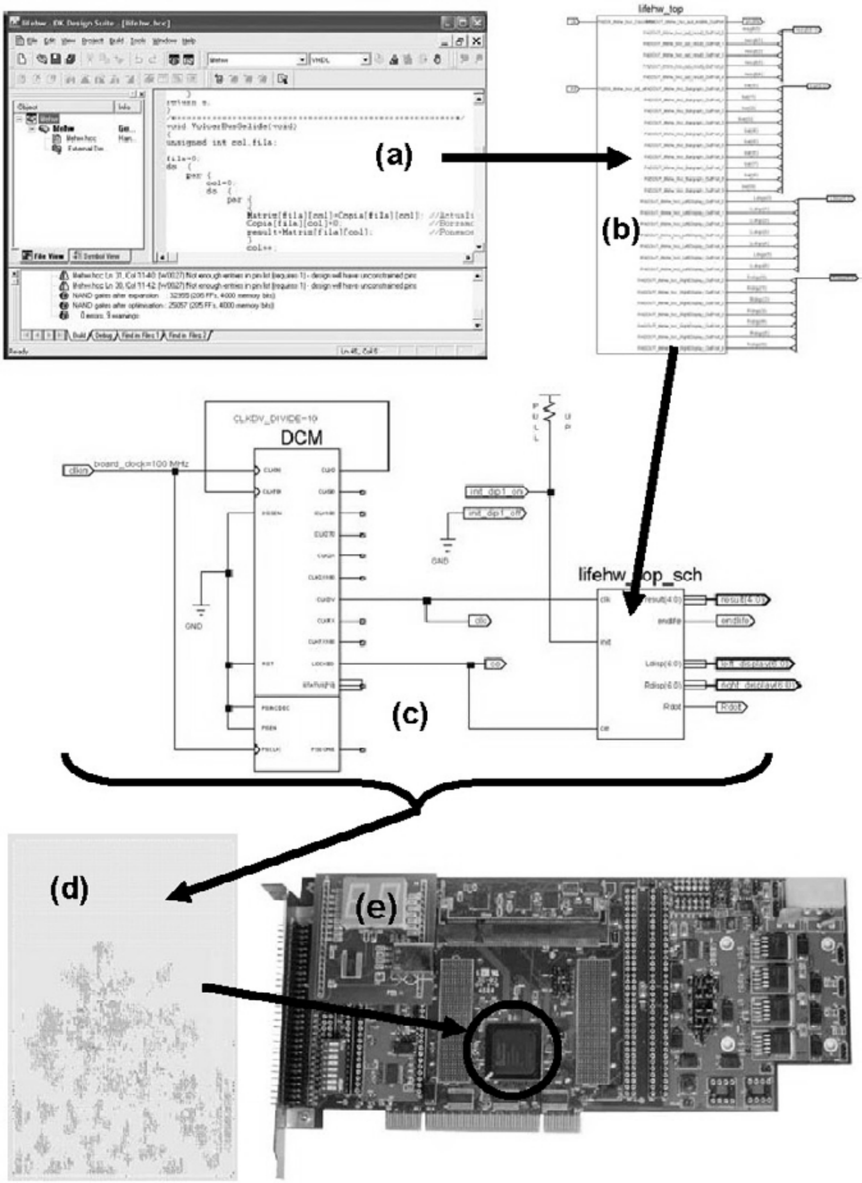
4.1. Modeling of the Algorithm

The first stage of the process consists of the modeling of the CA using high-level hardware description languages. For this purpose we have used Handel-C. This language allows us a hardware programming similar to the C language programming, but with real parallel programming capabilities. We have programmed two codes in Handel-C:

- Debug code. This code has the purpose of testing the algorithm to verify its performance before the hardware implementation. It generates a text file which is processed using an interpreter for watching the cells behavior in the screen.
- Hardware code (Figure 2.a). This code is based on the last one, modified in some few sentences. It generates a schematic symbol (Figure 2.b) for using in the Xilinx ISE design tool [11], as part of the controller used in the top schematic sheet of the CA processor (Figure 2.c).

4.2. Processor Design

In this stage the processor is designed according to the prototyping board features that it will be used. In this way, elements like switches, graphic displays, led, etc must be designed (Figure 2.c) in the top level view of the final processor schematic sheet. Also, the clock signal coming from the oscillator mounted on the board should be adapted by means of a clock manager to suit it to the timing restrictions of the synthesized circuit.



```
    }
return n;
}
/***************************************************************/
void VolverDesdeSlide(void)
{
unsigned int col,fila;

fila=0;
do  {
    par {
          col=0;
          do  {
                par {
                      }
                      Matriz[fila][col]=Copia[fila][col];  //Actuali
                      Copia[fila][col]=0;                  //Borrado
                      result=Matriz[fila][col];            //Ponemos
                      }
                      col++;
```

```
lifehw.hcc Ln 31, Col 11-40 [W0027] Not enough entries in pin list (requires 1) - design will have unconstrained pins
lifehw.hcc Ln 30, Col 11-42 [W0027] Not enough entries in pin list (requires 1) - design will have unconstrained pins
NAND gates after expansion : 32355 (205 FF's, 4000 memory bits)
NAND gates after optimisation : 25057 (205 FF's, 4000 memory bits)
    0 errors, 9 warnings
```

Figure 2. The process to develop the custom reconfigurable processor consists of five stages

4.3. Hardware Implementation

During the synthesis process, detailed and valuable information is obtained (Figure 2.d): occupied resources of the FPGA, maximum frequency allowed for clock source, power consumption, placed and routed layout, etc. After synthesis, the FPGA configuration file is obtained. This file is used to download the configuration bit-stream onto the FPGA mounted on the prototyping board (Figure 2.e) through a communication cable. Once the download is done, the board is ready to begin to be used as a CA custom processor.

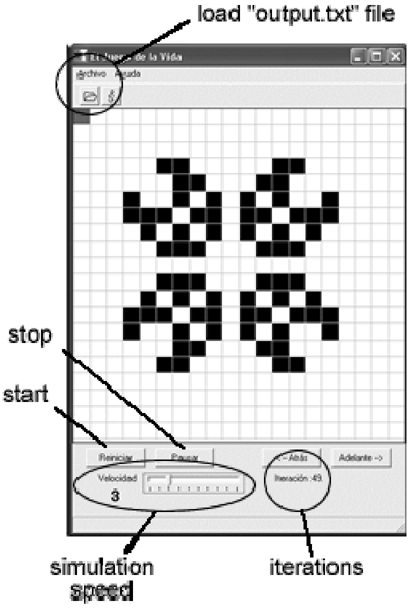


Figure 3. Graphical tool reading the data file generated from the software version of the CA to show its evolution

Table 2. Information related to the synthesis of the CA processor

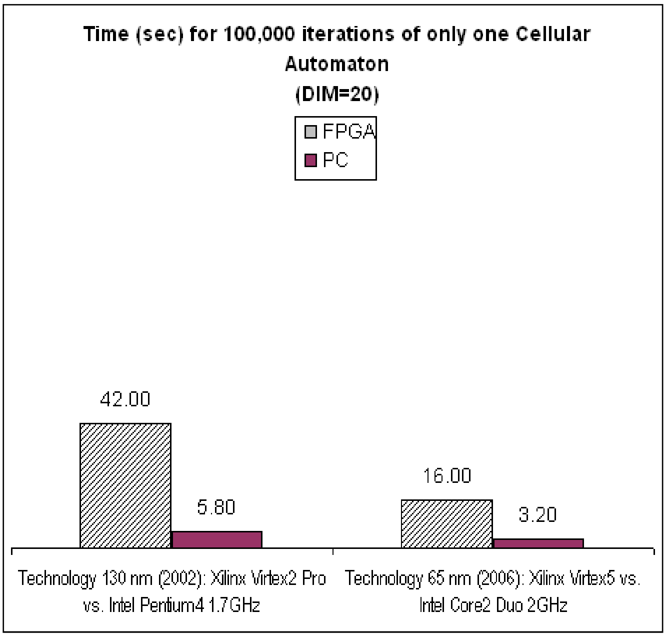
FPGA device	Xilinx Virtex2 Pro xc2vp30-7ff896		Xilinx Virtex5 LT330 xc5vlx330-1ff1760	
Cellular Automaton size	DIM=20	DIM=30	DIM=20	DIM=30
Maximum frequency allowed	12 MHz	14 MHz	27 MHz	31 MHz
On-board oscillator frequency	10 MHz	10 MHz	25 MHz	25 MHz
Occupied resources	25 %	37 %	3 %	5 %
Max. number of parallel CA on chip	4	2	33	20

5. Experimental Results

Table 2 shows the more important information reported during the synthesis process of the hardware design. This information is used, among other things, to select the frequency of the oscillator mounted on the prototyping board, to show the power consumption, and to estimate the maximum number of parallel CA running on chip, according to the occupied resources of the FPGA device. The elapsed time for the 100,000 iterations of the reconfigurable hardware processor has been measured by means of led signals.

In order to evaluate the real performance of the processor, a software version of the CA has been developed. This software consists of a C code exactly carrying out the same operations described in the Handel-C code used for hardware modeling, and optimized to reach the maximum possible speed. The obtained executable file has been executed on the two personal computers shown in Table 1. We can see in Figure 3 a graphical tool that displays the output results of the software. This utility is not the software we use to compare with the reconfigurable processor, because it spends a lot of time writing data results to the hard disk, but it is a useful tool for the user to see the CA evolution.

The result of the time analysis for both implementations (hardware and software) establishes the real performance and the effectiveness of the hardware implementation.



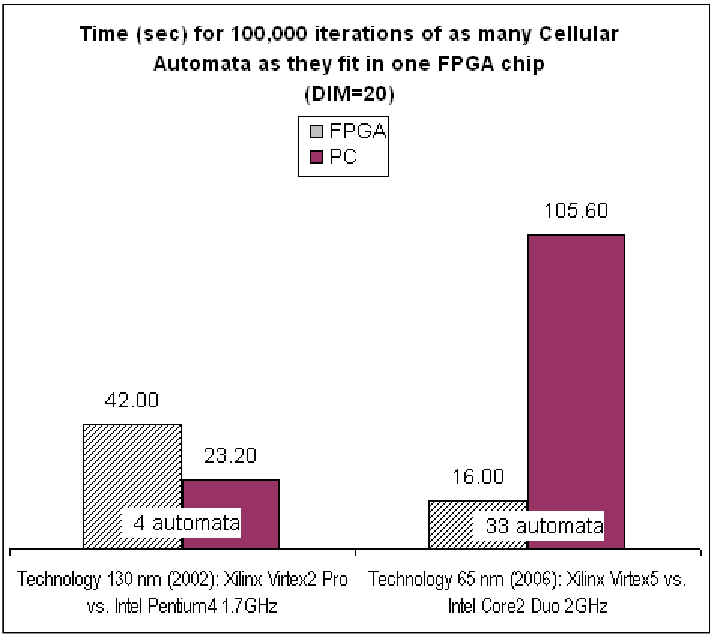
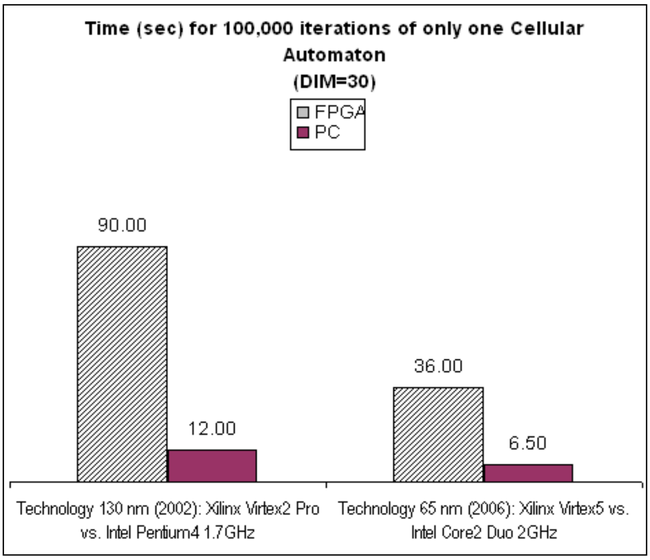


Figure 4. (Continued)



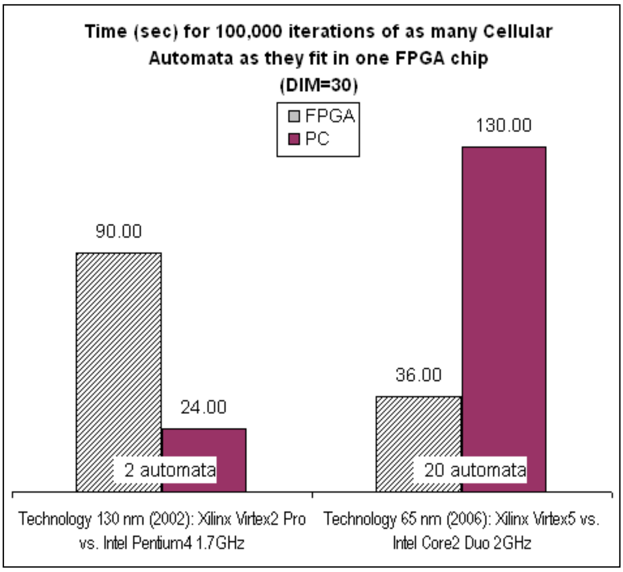


Figure 4. Summary of the time analysis obtained from the custom reconfigurable processor and from the software version for the different FPGA devices and computers considered. In all the cases the CA has executed 100,000 iterations. The graphs on the top are for DIM=20 and below for DIM=30. Also, the graphs on the left side correspond to one CA, since the graphs on the right side correspond to several CA running in parallel

In Figure 4 the summary of the time analysis is shown. The measures are compared between FPGA devices (running the CA hardware processor) and general-purpose processors (running the software version) of similar technologies (as CMOS features and year of launch), as it is described in Table 1. Two cases of CA sizes have been considered (DIM=20 and DIM=30), where the CA executes 100,000 iterations anyway. The graphs on the top are for DIM=20 and below for DIM=30. Also, the graphs on the left side correspond to a CA running alone, since the graphs on the right side correspond to several CA running in parallel.

When considering the case of more than one CA running in parallel, we calculate the maximum number of CA able to fit the FPGA device, according to the reported occupied resources during the synthesis phase. This way, the software version executes the 100,000 iterations of the CA multiplied by the same number of parallel CA in the FPGA device, in order to do a realistic comparison.

Analyzing the graphs we can verify that the software solution is clearly more advantageous for the older FPGA technology, even considering parallel CA, for one reason: the FPGA processor design does not enclose a high level of parallelism in its operations. Nevertheless, for the newer FPGA technology used, the hardware results are better than software results in all cases (although the efficiency slightly reduces when the CA size increases).

There are many methods to increase more the FPGA performance. A secure way is to update the cells of the CA at the end of the iteration in parallel, although we have not yet proven this possibility. Also, we can increase the clock frequency modifying the processor architecture, optimizing the synthesis parameters and considering FPGA devices with a larger speed grade than the used one.

The most effective way to increase a lot the performance is to distribute many parallel CA in more than one FPGA device. Pursuing this idea, the performance linearly increases. For example, if we consider a prototyping board mounting four Xilinx Virtex5 LT330 FPGA devices, we can put to work 33 CA in each one (if DIM=20), so the whole system could have up to 33 x 4=132 CA running in parallel, executing the 100,000 iterations in 16 seconds. At the same time, an Intel Core2 Duo E4500 2.2 GHz processor would need to execute 100,000 x 132 = 13,200,000 iterations to emulate the reconfigurable hardware platform, spending 3.2 seconds x 132 = 7 minutes. Taking into account the very low power consumption of the FPGA devices (less than 1 watt) in relation to the general-purpose processors (around 100 watt), a multi-FPGA board emerges as a low cost and high performance computing platform for running CA in intensive computing scenarios.

6. Conclusions and Future Work

The interest of a hardware solution based on FPGAs lies in the possibility of accelerating algorithms where many Cellular Automata can run in parallel inside FPGA devices and the possibility of freeing the computer resources (which otherwise would be dedicated almost exclusively to computing the solution to the problem over too great a period of time). The low cost, low power consumption and high timing performance offered by the FPGA solution can compete even against general-purpose clusters.

For this reason, the possibility to have CA processing in a stand-alone way by means of a specifically designed processor for this purpose is the sufficiently interesting thing as to explore in depth this computational alternative. In this sense, our forthcoming research is to connect two or more FPGA devices to process algorithms in parallel using Cellular Automata as basic element, increasing this way the overall performance. This research line can offer better performance results where the reconfigurable processor overcomes anyway to the computer.

ACKNOWLEDGMENTS

This work has been partially funded by the Spanish Ministry of Education and Science and FEDER under contract TIN2008-06491-C04-04 (the MSTAR project).

REFERENCES

[1] El Dessouki, W., Fathi, Y. & Rouphail, N. (2001). Meta-Optimization Using Cellular Automata with Application to the Combined Trip Distribution and Assignment System Optimal Problem, *Computer-Aided Civil and Infrastructure Engineering*, *16*, 384-398.

[2] Tovar, A., Patel, N., Kaushik, A. & Renaud, J. (2007). Optimality Conditions of the Hybrid Cellular Automata for Structural Optimization, *Journal of the American Institute of Aeronautics and Astronautics, 45*, 673-683.

[3] Qian, F. & Hirata, H. (1996). A Parallel Learning Cellular Automata for Combinatorial Optimization Problems, in *Proc. of IEEE International Conference on Evolutionary Computation,* 553-558.

[4] Gardner, M. (1970). Matematical Games: The Fantastic Combinations of John Conway's New Solitaire Game of Life, *Scientific American, 223*, 120–123.

[5] Hauck, S. & DeHon, A. (eds.), (2008). Reconfigurable Computing, The Theory and Practice of FPGA-Based Computation (Morgan Kaufmann).

[6] Vega, M. A., Sánchez, J. M. & Gómez, J. A. (2005). Advances in FPGA Tools and Techniques, *Microprocessors and Microsystems*, *29*, 47–50.

[7] Gajski, D., Jianwen, Z., Dömer, R., Gerstlauer, A. & Shuqing, Z. (2000). *SpecC: Specification Language and Methodology* (Springer-Verlag, New York).

[8] Arnout, G. (2000). *SystemC, in Proc. of the 2000 ACM Conference on Asia South Pacific Design Automation*, 573-578.

[9] Vernalde, S., Schaumont, P. & Bolsens, I. (1999). An Object Oriented Programming Approach for Hardware Design, in *Proc. of the IEEE Computer Society Workshop on VLSI'99* (IEEE Press), 68.

[10] Ramamritham, K. & Arya, K. (2004). System Software for Embedded Applications, in *Proc. 17th IEEE International Conference on VLSI Design* (IEEE Press), 22.

In: Cellular Automata
Editor: Thomas M. Li, pp. 91-117

ISBN: 978-1-61761-592-4

Chapter 6

EVOLVING CELLULAR AUTOMATA FOR FORM GENERATION IN ARTIFICIAL DEVELOPMENT

Arturo Chavoya[*]
Universidad de Guadalajara, Periférico Norte 799-L308,
Zapopan, Jal., México CP 45100

Abstract

Form generation or morphogenesis has a crucial role in both artificial and natural development. This chapter presents results from simulations in which a genetic algorithm (GA) was used to evolve cellular automata (CA) in order to generate predefined 2D and 3D shapes. The 2D shapes initially considered were a square, a diamond, a triangle and a circle, whereas for the 3D case the shapes chosen were a cube and a sphere. The CA's rule was defined as a lookup table where the input was determined by the interaction neighborhood's cell state values, and the output established whether or not a cell was to be reproduced at the empty objective cell. Four different 2D interaction neighborhoods were considered: von Neumann, Moore, 2-Radial, and Margolus; a 3D Margolus neighborhood was used to generate the sphere and the cube. In all cases, the GA worked by evolving the chromosomes consisting of the CA rule table's output bits and a section of bits coding for the number of iterations that the model was to run. After the final chromosomes were obtained for all shapes, the CA model was allowed to run starting with a single cell in the middle of the lattice until the allowed number of iterations was reached and a shape was formed. The transition rules that formed some of these basic shapes and others were later combined with an Artificial (gene) Regulatory Network (ARN) to make up genomes that controlled the activation sequence of the CA's rules to generate predefined patterns. The ARN was also evolved by a GA in order to produce cell patterns through the selective activation and inhibition of genes. Morphogenetic gradients were used to provide cells with positional information that constrained cellular replication. After a genome was evolved, a single cell in the middle of the CA lattice was allowed to reproduce until a desired cell pattern consisting of the combination of basic forms was generated.

Keywords: cellular automata, form generation, artificial development, genetic algorithm, artificial regulatory network

[*]E-mail address: achavoya@cucea.udg.mx

1. Introduction

Artificial development is the study of computer models of cellular growth, with the objective of understanding how complex structures and forms can emerge from a small group of undifferentiated initial cells. In biological systems, development is a fascinating and very complex process that involves following an extremely intricate program coded in the organism's genome. To present day, we still marvel at how from a single initial cell —the zygote— a whole functional organism of trillions of coordinated cells can emerge.

One of the crucial stages in the development of an organism is that of form generation or morphogenesis, where the fundamental body patterns of the individual are generated. It is now evident that gene regulatory networks play a central role in the development and metabolism of living organisms [1]. It has been discovered in recent years that the diverse cell patterns created during the developmental stages are mainly due to the selective activation and inhibition of very specific regulatory genes.

Artificial Regulatory Networks (ARNs) are computer models that seek to emulate the gene regulatory networks found in nature. ARNs have previously been used to study differential gene expression either as a computational paradigm or to solve particular problems [2, 3, 4, 5, 6]. On the other hand, evolutionary computation techniques have been extensively used in the past in a wide range of applications, and in particular they have previously been used to evolve ARNs to perform specific tasks [7, 8].

Over the years, artificial models of cellular growth have been proposed with the objective of understanding the intricacies of the development process, including some that involve the use of cellular automata [9, 10, 11, 12, 13, 14]. In this chapter an artificial development model that generates cellular patterns in 2D and 3D by means of the selective activation and inhibition of development genes under the constraints of morphogenetic gradients is presented. Cellular growth is achieved through the expression of structural genes, which are in turn controlled by an ARN evolved by a Genetic Algorithm (GA). The ARN establishes the time at which cells can reproduce and determines which structural gene to use at each time step. At the same time, morphogenetic gradients constrain the position at which cells can replicate. The combination of the ARN and the structural genes make up the artificial cell's genome.

In order to test the functionality of the ARN found by the GA, a cellular growth testbed based on the Cellular Automata (CA) paradigm was first developed, so that the GA chromosomes representing the proposed ARN models could be evaluated in their role to produce the desired patterns [15]. Cellular automata have previously been used to study form generation, as they provide an excellent framework for modeling local interactions that give rise to emergent properties in complex systems [16, 17].

2. Cellular Growth Testbed

In order to evaluate the performance of the development programs obtained in the the 2D and 3D models, their evolved genomes were applied to a cellular growth testbed designed to generate simple geometrical shapes [15]. This growth model is based on the extensively studied CA paradigm.

Cellular automata are simple mathematical models that can be used to study self-organization in a wide variety of complex systems [18]. CA are characterized by a regular lattice of N identical cells, an interaction neighborhood template η, a finite set of cell states Σ, and a space- and time-independent transition rule ϕ which is applied to every cell in the lattice at each time step [17].

In the CA models presented in this chapter, a cell can become active only if there is already an active cell in the interaction neighborhood. Thus, a new active cell can only be derived (reproduced) from a previously active cell in the interaction neighborhood, i.e. no spontaneous generation is allowed, as in actual biological systems.

Two different regular lattices with non-periodic boundaries were tried, a 2D and a 3D lattice. For 2D neighborhoods, a 33×33 cell lattice was used, whereas for the 3D neighborhood a cubic lattice of side length 17 was chosen. The set of cell states was defined as $\Sigma = \{0, 1\}$, where 0 can be interpreted as an empty cell and 1 as an occupied or active cell. For 2D shapes, four different interaction neighborhood templates η were considered, while only one neighborhood was studied in 3D. The interaction neighborhoods are described in the following subsections. The CA's rule ϕ was defined as a lookup table that determined, for each local neighborhood, the state (empty, occupied) of the objective cell at the next time step [19, 20]. For a binary-state CA, these update states are termed the rule table's "output bits". The lookup table input was defined by the binary state value of cells in the local interaction neighborhood, where 0 meant an empty cell and 1 meant an occupied cell.

All the neighborhoods considered in the cellular growth testbed are *outer* interaction neighborhoods, since the objective cell is not considered as part of the interaction neighborhood. This simplification was made given that the CA rule is applied only to empty cells, implicitly assuming that all rules that have the state value 1 in the objective cell also have the value 1 as output. That is, a cell that is already occupied by an active cell has no place to hold another active cell.

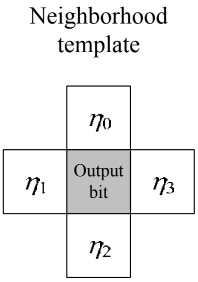
Figure 1 shows an example of the relationship between a CA neighborhood template and the corresponding lookup table. For each neighborhood configuration, the output bit determines whether or not a cell is to be placed at the corresponding objective cell position. In this example, if there is only an active cell at the objective cell's right position, then the objective cell is to be filled with an active cell (second row of the lookup table in Fig. 1). The actual output bit values used have to be determined for each different shape and are found using a genetic algorithm.

2.1. 2D Neighborhoods

Four types of 2D interaction neighborhoods were used: von Neumann, Moore, 2-Radial, and Margolus neighborhoods (Fig. 2).

2.1.1. Von Neumann Neighborhood

In the von Neumann neighborhood the cells at the top, left, bottom, and right of the objective cell make up the interaction neighborhood. The CA lookup table input ϕ is defined by the binary values of cells $\eta_0^v \eta_1^v \eta_2^v \eta_3^v$ of the neighborhood indicated in Fig. 2(a).



Lookup Table

η_0	η_1	η_2	η_3	Output bit
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
	⋮			⋮
1	1	1	1	0

Figure 1. Relationship between a cellular automaton neighborhood template and the corresponding lookup table. The output bit values shown are only for illustration purposes.

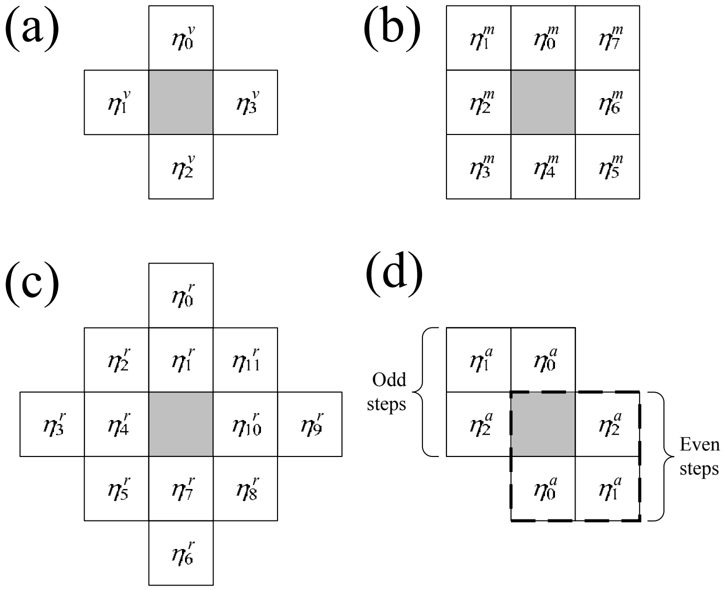


Figure 2. 2D interaction neighborhoods. (a) Von Neumann. (b) Moore. (c) 2-Radial. (d) Margolus. The objective cell is depicted in gray.

2.1.2. Moore Neighborhood

In the Moore neighborhood the nearest eight cells around the objective cell define the interaction neighborhood. The lookup table input is defined by the binary values of $\eta_0^m\eta_1^m\eta_2^m\eta_3^m\eta_4^m\eta_5^m\eta_6^m\eta_7^m$ of the neighboring cells shown in Fig. 2(b).

2.1.3. 2-Radial Neighborhood

This interaction neighborhood is composed by all the cells within a radius of two cells of length from the objective cell. Formally,

$$\eta^r = \left\{(c_x, c_y) : c_x, c_y \in \{-2,-1,0,1,2\} \wedge \sqrt{c_x^2 + c_y^2} \leq 2\right\} - \{(0,0)\}$$

The lookup table input is defined by the binary values of $\eta_0^r\eta_1^r\eta_2^r\eta_3^r\eta_4^r\eta_5^r\eta_6^r\eta_7^r\eta_8^r\eta_9^r\eta_{10}^r\eta_{11}^r$, where η_0^r to η_{11}^r are as indicated in Fig. 2(c).

2.1.4. Margolus Neighborhood

In the Margolus neighborhood there is an alternation of the block of cells considered at each step of the CA algorithm. At odd steps, the cells at the top, upper left, and left of the objective cell constitute the interaction neighborhood, while at even steps the neighborhood is formed by the mirror cells of the previous block (see Fig. 2(d)). The lookup table input is defined by the binary values of $\varsigma\eta_0^a\eta_1^a\eta_2^a$, where ς is defined as 0 for odd steps of the CA algorithm and as 1 for even steps, and η_0^a, η_1^a and η_2^a are the binary values of the neighboring cells indicated in Fig. 2(d).

2.2. 3D Neighborhood

The template chosen for working in 3D was the Margolus neighborhood, which has been previously used with success in modeling 3D shapes [21, 22]. As in the 2D case, each cell belongs to different blocks at odd and even steps. The lookup table input is defined by the values of $\varsigma\eta_0''\eta_1''\eta_2''\eta_3''\eta_4''\eta_5''\eta_6''$, where ς is defined as in the 2D case and η_0'' to η_6'' are as indicated in Fig. 3.

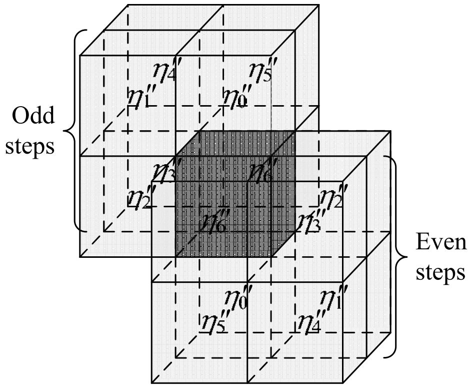


Figure 3. 3D Margolus neighborhood. The objective cell is depicted as a darker cube.

2.3. NetLogo Models

NetLogo is a programmable modeling environment based on StarLogo that can be used to simulate natural and social phenomena [23]. It works by giving instructions to hundreds or thousands of independent “agents” all operating concurrently. It is well suited to study emergent properties in complex systems that result from the interaction of simple but often numerous entities.

For all simulations, the CA algorithm at study was implemented as a NetLogo model. For each of the neighborhoods studied and for each of the models proposed, a NetLogo model was built. Each cell position is defined by its Cartesian coordinates with the origin at the center of the lattice. Starting with an active cell in the middle of the lattice, the CA algorithm was applied allowing active cells to reproduce according to the CA rule table and until the indicated number of iterations was attained.

3. Morphogenetic Gradients

Ever since Turing's influential article on the theoretical effect of diffusing chemical substances on an organism's pattern development [24], the role of these molecules has been confirmed in a number of biological systems. These organizing substances have been termed *morphogens* due to their role in driving morphogenetic processes. In the development models presented in this chapter, morphogenetic gradients were generated similar to those found in the eggs of the fruit fly *Drosophila*, where orthogonal gradients offer a sort of Cartesian coordinate system [25]. These gradients provide reproducing cells with positional information in order to facilitate the spatial generation of patterns. The artificial morphogenetic gradients were set up as suggested in [10], where morphogens diffuse from a source towards a sink, with uniform morphogen degradation throughout the gradient.

Before cells were allowed to reproduce in the cellular growth testbed, morphogenetic gradients were generated by diffusing the morphogens from one of the CA boundaries for 1000 time steps. Initial morphogen concentration level was set at 255 arbitrary units, and the source was replenished to the same level at the beginning of each cycle. The sink was set up at the opposite boundary of the lattice, where the morphogen level was always set to zero. At the end of each time step, morphogens were degraded at a rate of 0.005 throughout the CA lattice.

For the 2D model, two orthogonal gradients in the CA lattice were defined, one generated from left to right and the other from top to bottom (Fig. 4).

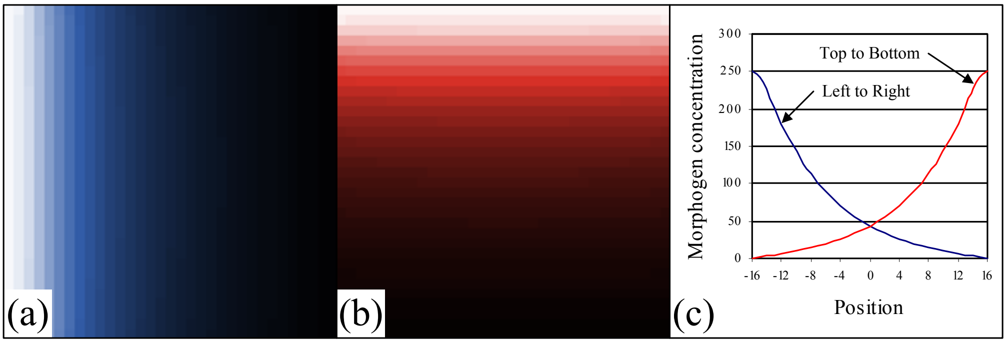


Figure 4. 2D morphogenetic gradients. (a) Left to Right. (b) Top to Bottom. (c) Morphogen concentration graph.

In the case of the 3D model, three orthogonal gradients were defined in the CA lattice, one for each of the main Cartesian axes (Fig. 5). In the 3D figures presented in this chapter

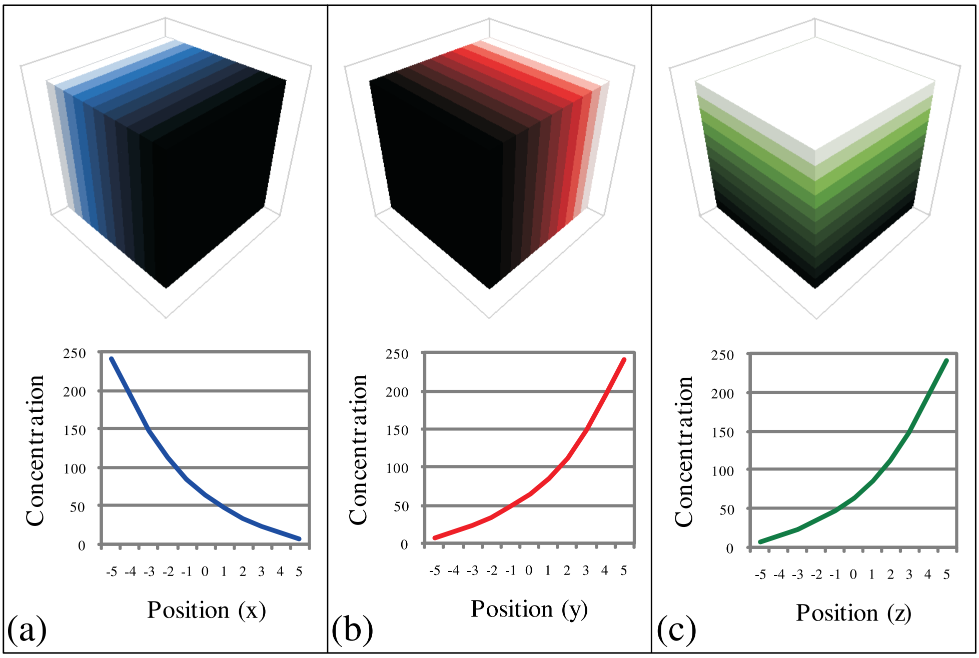


Figure 5. 3D morphogenetic gradients. Positions with highest morphogen concentration are depicted in white; darker tones mean lower concentrations. **(a)** Left to right (*x-axis*). **(b)** Back to front (*y-axis*). **(c)** Top to bottom (*z-axis*).

the following conventions are used: in the 3D insets the positive *x*-axis extends to right, the positive *y*-axis is towards de back of the page, the positive *z*-axis points to the top, and the axes are rotated 45 degrees to the left to show a better perspective.

4. Genomes

Genomes are the repository of genetic information in living organisms. They are encoded as one or more chains of DNA, and they regularly interact with other macromolecules, such as RNA and proteins. Artificial genomes are typically coded as strings of discrete data types. The genomes used in the following models were defined as binary strings starting with a series of regulatory genes, followed by a number of structural genes.

Two different artificial genomes are proposed in this chapter, one for 2D and one for 3D. In both cases, genomes were defined as binary strings starting with a series of ten regulatory genes that constitutes an artificial regulatory network, followed by a series of structural genes, which contain the CA's lookup tables that control cell reproduction.

In the first genome, each regulatory gene consists of a series of eight inhibitor/enhancer sites, a series of five regulatory protein coding regions, and two morphogen threshold activation sites that determine the allowed positions for cell reproduction (Fig. 6). Inhibitor/enhancer sites are composed of a 12-bit function defining region and a regulatory site. The parameter values used for the number of inhibitor/enhancer sites and the number of function defining bits are those that gave the best results under the conditions tested [26].

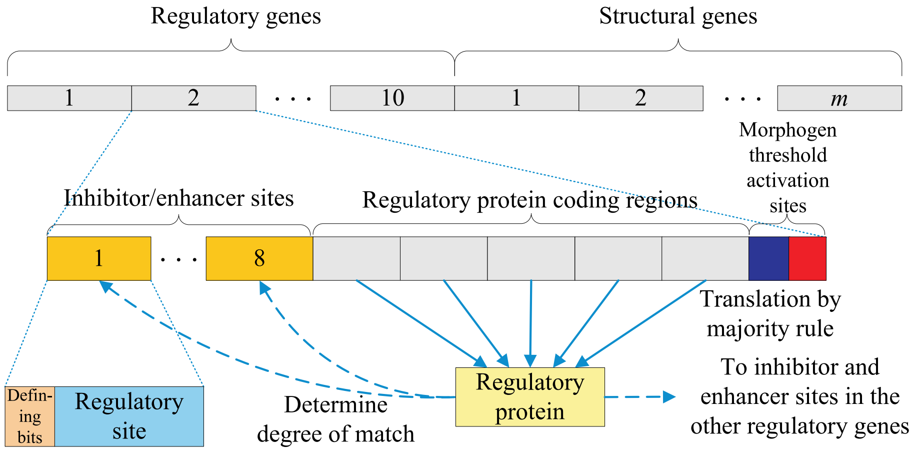

Regulatory genes
Structural genes
1
2
· · ·
10
1
2
· · ·
m
Morphogen threshold activation sites
Inhibitor/enhancer sites
Regulatory protein coding regions
1
· · ·
8
Translation by majority rule
To inhibitor and enhancer sites in the other regulatory genes
Regulatory protein
Determine degree of match
Defin-ing bits
Regulatory site

Figure 6. Genome structure and regulatory gene detail for the 2D model. Regulatory genes make up an artificial regulatory network, whereas structural genes contain the lookup tables that control cell reproduction.

The genome for the 3D model is an extension of the first genome and its structure is shown in Fig. 7. The difference lies in the incorporation of an additional morphogen activation site at the end of each regulatory gene to account for the *z*-axis.

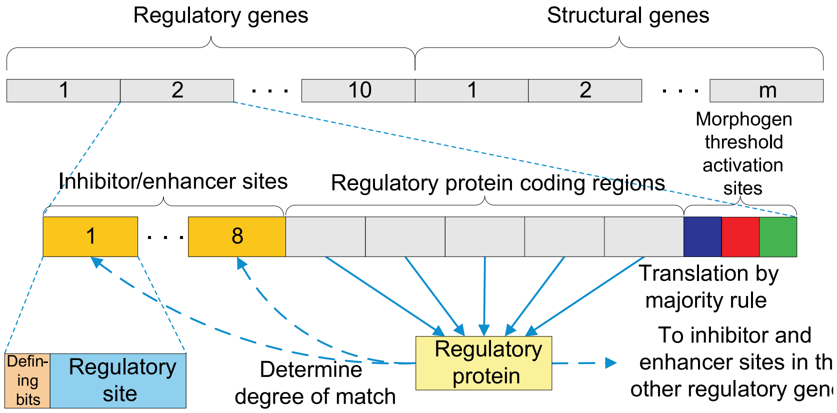

Regulatory genes
Structural genes
1
2
· · ·
10
1
2
· · ·
m
Morphogen threshold activation sites
Inhibitor/enhancer sites
Regulatory protein coding regions
1
· · ·
8
Translation by majority rule
To inhibitor and enhancer sites in th
other regulatory gen
Regulatory protein
Determine degree of match
Defin-ing bits
Regulatory site

Figure 7. Genome structure and regulatory gene detail for the 3D model. A morphogen threshold activation site was added at the end of each regulatory gene.

Structural genes are always associated with the corresponding regulatory genes, that is, structural gene number 1 is associated with regulatory gene number 1 and its related translated protein, and so on. A structural gene was defined as being active if and only if the regulatory protein translated by the associated regulatory gene was above a certain concentration threshold. The value chosen for the threshold was 0.5 as the sum of all protein concentrations is always 1.0, making it impossible for two or more regulatory proteins to be with a concentration above 0.5 units at the same time. As a result, one structural gene at most can be expressed at a particular time step in a cell. A structural gene is interpreted as a CA rule table by reading its bits as output bits of the CA rule. If a structural gene is active, then the CA lookup table coded in it is used to control cell reproduction. If no protein concentration is above the 0.5 threshold, then cell reproduction cannot occur.

The gene regulatory networks presented in this chapter are an extension of the model originally proposed by Banzhaf [4]. However, unlike the ARN developed by this author, genes implemented in the present models are not preceded by promoter sequences and there are no unused intergene regions. Promoters in biology indicate where regulatory binding sites begin. In Banzhaf's model, a genome is a randomly generated bit string where the beginning of a gene is signaled by a fixed arbitrary 8-bit sequence. As a result, this promoter sequence can occur with a probability of $2^{-8} \approx 0.0039 = 0.39\%$ and the number of genes is proportional to the length of the genome.

For the present model, it was decided that unused bit sequences between genes would be a waste of space and an additional source of variation in the evolution experiments, since the number of regulatory genes could vary from genome to genome. Furthermore, this approach would make relating a fixed number of structural genes to a varying number of regulatory genes difficult. As a result it would be possible to have less regulatory genes than structural genes, which went against the concept of one regulatory gene controlling one structural gene.

In the two models presented, all regulatory genes are adjacent and have predefined initial and end positions. Furthermore, the number of regulatory genes is fixed and the number of regulatory sites is more than two and they can behave either as an activator or an inhibitor, depending on the configuration of the function defining bits associated with the regulatory site [26]. If there are more 1s than 0s in the function defining region, then the site functions as an activator, but if there are more 0s than 1s, then the site behaves as an inhibitor. Finally, if there is an equal number of 1s and 0s, then the regulatory site is turned off. This means that the regulatory site role as an activator or as an inhibitor can be evolved by the GA. Furthermore, if the number of function defining bits is even, then the regulatory site can be turned on and off. The number of regulatory sites was extended with respect to the original model in order to more closely follow what happens in nature, where biological regulatory genes involved in development typically have several regulatory sites associated with them [1].

In addition to the inhibitor/activator sites, each regulatory gene contains a series of five regulatory protein coding regions which "translate" a protein using the majority rule, i.e. for each bit position in the protein coding regions, the number of 1s and 0s is counted and the bit that is in majority is translated into the regulatory protein. An odd number of regulatory protein coding regions sites was chosen by Banzhaf in order to avoid ties when applying the majority rule.

The regulatory sites and the individual protein coding regions all have the same size of 32 bits. Thus the protein translated from the coding regions can be compared on a bit by bit basis with the regulatory sites from the inhibitors and activators, and the degree of matching can be measured. As in [4], the comparison was implemented by an XOR operation, which results in a "1" if the corresponding bits are complementary.

Each translated protein is compared with the inhibitor and activator sites of all the regulatory genes in order to determine the degree of interaction in the regulatory network. The influence of a protein on an activator or inhibitor site is exponential with the number of matching bits. The strength of excitation *en* or inhibition *in* for gene i with $i = 1, \ldots, n$ is defined as

$$en_i = \frac{1}{v}\sum_{j=1}^{v} c_j e^{\beta\left(u_{ij}^{+} - u_{\max}^{+}\right)} \text{ and} \tag{1}$$

$$in_i = \frac{1}{w}\sum_{j=1}^{w} c_j e^{\beta\left(u_{ij}^{-} - u_{\max}^{-}\right)} , \tag{2}$$

where n is the total number of regulatory genes, v and w are the total number of activator and inhibitor sites, respectively, c_j is the concentration of protein j, β is a constant that fine-tunes the strength of matching, u_{ij}^{+} and u_{ij}^{-} are the number of matches between protein j and the activator and inhibitor sites of gene i, respectively, and $u_{\max}^{+}$ and $u_{\max}^{-}$ are the maximum matches achievable between a protein and an activator or inhibitor site, respectively [4].

Once the *en* and *in* values are obtained for all regulatory genes, the corresponding change in concentration c for protein i in one time step is found using

$$\frac{dc_i}{dt} = \delta\left(en_i - in_i\right) c_i , \tag{3}$$

where δ is a constant that regulates the degree of protein concentration change. Parameters β and δ were set to 1.0 and 1.0×10^6, respectively, as previously reported [27].

Protein concentrations are updated and if a new protein concentration results in a negative value, the protein concentration is set to zero. Protein concentrations are then normalized so that total protein concentration is always the unity. At time step 0, all ten proteins start out with the same concentration level, i.e. with a value of 0.1 units.

As for structural genes, they code for the particular shape grown by the reproducing cells and were obtained using the methodology presented in [15]. Briefly, in order to produce predefined 2D and 3D shapes, a gene was evolved by a GA in the cellular growth testbed described in Section 2. The GA worked by evolving the CA rule table's output bits.

In the series of simulations presented in this chapter, the number of structural genes is always less than the number of regulatory genes. Thus, some proteins both regulate concentration for other proteins and directly control structural gene expression, while others only have a regulatory role. Structural gene expression is visualized in the cellular growth testbed as a distinct external color for the cell. Thus, cells with different external color represent differentiated cells that express a specific structural gene. The color associated with a structural gene is assigned to a cell when it is created as a result of the activation of that particular structural gene.

The morphogen threshold activation sites are the last elements of the regulatory gene and they can provide reproducing cells with positional information as to where they are allowed to grow in the CA lattice. There is one site for each of the orthogonal morphogenetic gradients described in Section 3.. These sites are 9 bits in length, where the first bit defines the allowed direction (above or below the threshold) of cellular growth, and the next 8 bits code for the morphogen threshold activation level, which ranges from 0 to $2^8 - 1 = 255$. If the site's high order bit is 0, then cells are allowed to replicate below the morphogen threshold level coded in the lower order eight bits; if the value is 1, then cells are allowed to reproduce above the threshold level. Since in a regulatory gene there is one site for each of the orthogonal morphogenetic gradients, for each set of morphogen threshold activation levels, the high order bits define in which of the four relative quadrants or eight relative octants, cells expressing the associated structural gene can reproduce in the 2D or 3D model, respectively.

5. Genetic Algorithm

Genetic algorithms are search and optimization methods based on ideas borrowed from natural genetics and evolution [28]. A GA starts with a population of chromosomes representing vectors in search space. Each chromosome is evaluated according to a fitness function and the best individuals are selected. A new generation of chromosomes is created by applying genetic operators on selected individuals from the previous generation. The process is repeated until the desired number of generations is reached or until the desired individual is found.

The GA presented in this chapter uses tournament selection as described in [29] with single-point crossover and mutation as genetic operators. Single-point crossover consists of randomly selecting two chromosomes with a certain probability called crossover rate, and then randomly selecting a single bit position in the chromosome structure. From this point on, the remaining fragments of the two chromosomes are exchanged. The resulting chromosomes then replace the original ones in the chromosome population. On the other hand, mutation consists of randomly flipping one bit in a chromosome from 0 to 1 or vice versa. The probability of each bit to be flipped is called the mutation rate.

After several calibration experiments, the parameter values described next were considered to be appropriate. The initial population consisted of either 500 binary chromosomes chosen at random for evolving the form generating genes, or 1000 chromosomes for the simulations involving the ARN models. Tournaments were run with sets of 3 individuals randomly selected from the population. Crossover rate was 0.60 in all cases, whereas the mutation was 0.015 for the evolution of structural genes, and 0.15 for the evolution of ARNs. The crossover rate of 0.60 was chosen because it was reported to give the best results when trying to evolve a binary string representing a CA using a GA [30]. As for the mutation rate, it was decided to use a value one order of magnitude higher in the evolution of the ARN models than the one used in the same report because it was found that single bits could have a considerable influence on the final behavior of the ARN. In particular, in one simulation the flipping of a single bit almost doubled the fitness value of an evolving genome [27]. Finally, the number of generations was set at 50 in all cases, since there was no significant improvement after this number of generations.

5.1. Chromosome structure

The GA experiments were run with two different types of chromosomes, the kind used for the evolution of a form generating gene (structural genes in the artificial genomes), and the one used for evolving the ARN models.

Table 1. Size in bits for chromosomes used in evolving a form generating gene. Parameters are as defined in the text.

Neighborhood	l	a	b	$2^l \times a + b$
Von Neumann	4	1	4	20
Moore	8	1	4	260
2-Radial	12	1	4	4100
2D Margolus	3	2	5	21
3D Margolus	7	2	4	260

5.1.1. Chromosome structure for form generation

The chromosome structure used for evolving a form generating gene is shown in Fig. 8. The *control field* codes for the number of steps in base 2 that the CA algorithm is allowed to run, whereas the *action field* represents the CA lookup table's output bits in lexicographical order of neighborhood.

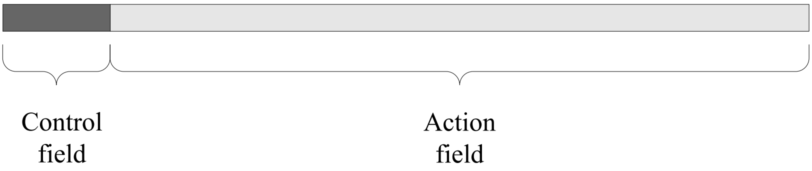


Figure 8. Chromosome structure for evolving a form generating gene.

For the initial simulations, when testing different neighborhood templates, chromosome size varied depending on the neighborhood type and the maximum number of iterations allowed for the CA. For this type of chromosome, size in bits was defined as

$$(2^l \times a) + b, \tag{4}$$

where l is the number of cells in the local interaction neighborhood (excluding the objective cell, which is always assumed to be 0, i.e. the CA rule is applied only to empty cells), a is the number of alternating steps in the CA algorithm (2 for the Margolus neighborhood and 1 for all the others), and b is the number of bits of the maximum number of iterations in base 2 that the CA is allowed to run. Table 1 shows the chromosome size for the various neighborhoods considered.

The control field size was chosen so that the shape formed on any CA run could not overflow the boundaries of the CA lattice. As mentioned in Section 2., the lattices consisted of a square of 33 × 33 and a cube of 17 × 17 × 17 cells, with the initial active cell at the central position. To have an active cell reach one of the lattice boundaries, the CA algorithm would be required to run for at least 16 steps in the square lattice, and 8 steps in the cubic lattice. On this ground, for all 2D neighborhoods, except 2D Margolus, the control field size was chosen to be 4, so that the CA algorithm would iterate for at most $2^4 - 1 = 15$

steps. For the Margolus neighborhoods, due to the alternation of the cell blocks forming the interaction neighborhood, the CA algorithm would require twice as many steps for an active cell to reach one of the lattice boundaries. For this reason, the chromosome's control field size was defined as 5 for the 2D Margolus neighborhood and 4 in the 3D Margolus case, so that the upper limit of iterations would be $2^5 - 1 = 31$ and $2^4 - 1 = 15$ steps, respectively.

The chromosome's action field coding for the CA rule table is of length $2^l \times a$. For the Moore neighborhood, that represents a rule space of $2^{256} \approx 10^{77}$, and for the 2-Radial neighborhood the rule space size is $2^{4096} \approx 10^{1233}$. In both cases the search space is far too large for any sort of exhaustive evaluation. And if we also take into account the bits introduced by the control field, the search space grows larger. Even for the smallest of the neighborhoods considered, the von Neumann and the 2D Margolus neighborhoods, the search space is not negligible, since it contains over one million possibilities.

In the case of the 3D CA, the 3D Margolus neighborhood was chosen over, for example, a 3D Moore neighborhood, since in the latter case the interaction neighborhood would consist of the nearest 26 cells, giving a CA lookup table of $2^{26} \approx 6.7 \times 10^7$ rows, as opposed to the $2^7 \times 2 = 256$ rows required by the 3D Margolus template.

5.1.2. Chromosome structure for pattern generation

When evolving the ARNs with the goal of synchronizing the expression of structural genes, the chromosomes used for the GA runs were simply the ARN chains themselves. Chromosome sizes were 5300 and 5390 bits for the 2D and 3D models, respectively.

The ARN binary string for the 3D model represents a search space of $2^{5390} \approx 3.6 \times 10^{1622}$ vectors. Evidently, search space grows exponentially with the size of regulatory genes. But even in the case of the ARN for the 2D model, the search space has a size of $2^{5300} \approx 2.9 \times 10^{1595}$, which is still too large to be explored deterministically. It should be evident that the search space for any of the ARN models considered is far too large for any method of exhaustive assessment. Therefore, the use of an evolutionary search algorithm for finding an appropriate synchronization of gene expression is amply justified.

5.2. Fitness function

As in the case of the chromosome structure, there were two different fitness functions used, depending on whether one o more structural genes were considered.

5.2.1. One structural gene

The fitness function for the simulations involving the evolution of one structural gene is the same as the function used by de Garis for evolving CA [16]:

$$Fitness = \frac{ins - \frac{1}{2}outs}{des}, \tag{5}$$

where *ins* is the number of filled cells inside the desired shape, *outs* is the number of filled cells outside the desired shape, and *des* is the total number of cells inside the desired shape. Thus, a fitness value of 1 represents a perfect match.

During the course of a GA experiment, each chromosome produced in a generation was fed to the corresponding NetLogo model, which was allowed to run for as many iterations as indicated in the chromosome's control field. Fitness was evaluated after the model stopped and a shape was formed. This process continued until the maximum number of generations was reached and then the best individual was selected.

5.2.2. Multiple structural genes

In the case of the evolution of ARNs that synchronized the expression of more than one structural gene, the fitness function used by the GA was defined as

$$Fitness = \frac{1}{c}\sum_{i=1}^{c}\frac{ins_i - \frac{1}{2}outs_i}{des_i}, \tag{6}$$

where c is the number of different colored shapes, each corresponding to an expressed structural gene, ins_i is the number of filled cells inside the desired shape i with the correct color, $outs_i$ is the number of filled cells outside the desired shape i, but with the correct color, and des_i is the total number of cells inside the desired shape i. In consequence, a fitness value of 1 represents again a perfect match. This fitness function is an extension of the one used in [16], where the shape produced by only one "gene" was considered. To account for the expression of several structural genes, the combined fitness values of all structural gene products were introduced in the fitness function used.

During a GA run, each chromosome produced in a generation was fed to the corresponding NetLogo model, where the previously evolved structural genes were attached and the cells were allowed to reproduce controlled by the ARN found by the GA. Fitness was evaluated at the end of 100 time steps in the cellular growth testbed, where a colored pattern could develop. This process continued until the maximum number of generations was reached or when an individual with a fitness value of 1 was obtained.

6. Form Generation

In all cases, the GA described in Section 5. was used to evolve the lookup table and the number of iterations for the desired shapes. Starting with one active cell in the middle of the CA lattice in the NetLogo model, cells were allowed to reproduce (sprout an active cell from a previously active cell) using the lookup table found by the GA and for as many iterations as indicated in the chromosome's control field. Since the CA algorithm used asynchronous updating with the order of reproduction of cells randomly selected, a particular shape and fitness could slightly change on different runs of the CA algorithm for the same chromosome. For this reason, fitness mean and standard deviation from 100 runs of the CA algorithm are reported for all final chromosomes.

6.1. 2D shapes

The desired shapes are shown in Figure 9. These shapes were chosen for their simplicity and familiarity. The square has a side length of 21 cells, the diamond has a length of 11 cells from the center to any of its corners, the triangle has a base and a height of 23 and 21 cells, respectively, and finally the circle has a radius of 11 cells.

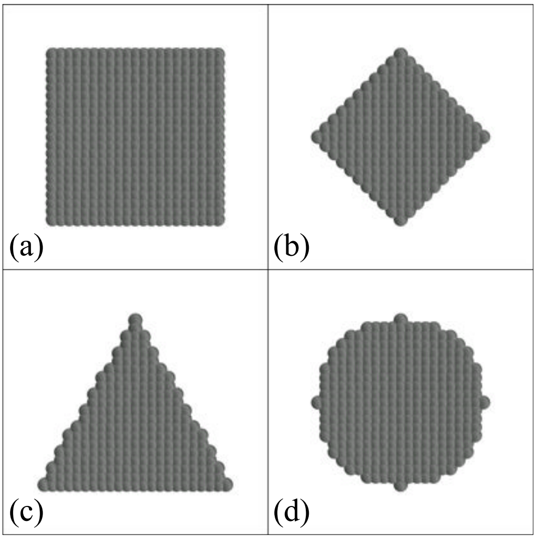


Figure 9. Desired shapes. (a) Square. (b) Diamond. (c) Triangle. (d) Circle.

Table 2 presents the fitness mean and standard deviation from 100 runs in the NetLogo model of the final chromosomes for all the shapes and neighborhoods considered. A comparative chart of the mean fitness values presented in Table 2 is shown in Figure 10, grouped by interaction neighborhood.

Table 2. Fitness mean ($\bar{x}$) and standard deviation (σ) from 100 runs of the CA algorithm for the final chromosomes.

	Von Neumann		**Moore**		**2-Radial**		**Margolus**	
Shape	$\bar{x}$	σ	$\bar{x}$	σ	$\bar{x}$	σ	$\bar{x}$	σ
Square	0.738	0.008	0.993	0.003	0.887	0.015	1.000	0.000
Diamond	1.000	0.000	0.805	0.040	0.773	0.028	0.880	0.018
Triangle	0.580	0.000	0.950	0.011	0.909	0.023	0.860	0.010
Circle	0.868	0.000	0.932	0.013	0.875	0.017	0.928	0.006
Average	**0.797**	**0.002**	**0.920**	**0.017**	**0.861**	**0.021**	**0.917**	**0.008**

At the initial stages of this work, it was assumed that, since all shapes considered had dimensions such that the outer active cells could be reached in 10 steps, the number of iterations should be fixed at 10 steps for the von Neumann, Moore and 2-Radial neighborhoods, and 20 steps for the Margolus neighborhood. However, it was later decided that, in order to avoid a preconceived notion of how the evolved chromosomes should work, the GA should also find the optimum number of iterations needed to generate a particular shape. For this reason the control field was introduced in the chromosome definition. Table 3 presents the evolved number of iterations (coded in the control field) of the final chromosomes for all shapes and neighborhoods.

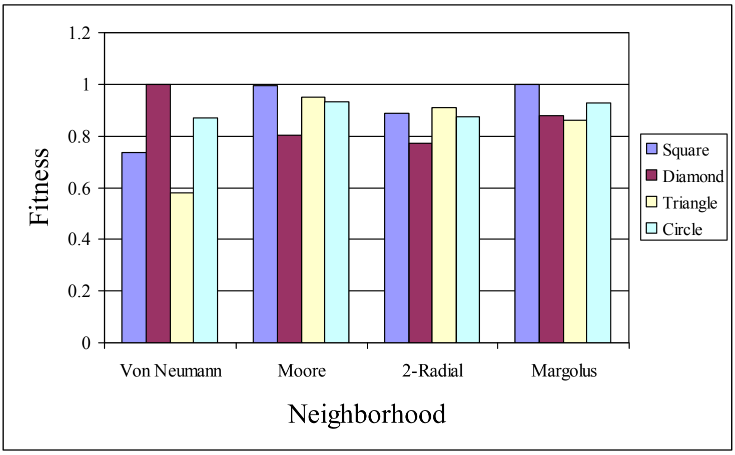


Figure 10. Mean fitness comparative chart for all neighborhoods and shapes.

Table 3. Evolved number of iterations for the final chromosomes for all neighborhoods and shapes.

Shape	Von Neumann	Moore	2-Radial	Margolus
Square	15	10	11	20
Diamond	10	12	9	19
Triangle	12	10	10	19
Circle	13	9	10	18

Figures 11 to 14 show results from some of the best runs for the four types of shapes obtained using the four models, corresponding to each of the neighborhoods studied. For ease of visualization, cells that fall outside the desired shape are shown in light gray [31].

6.2. 3D shapes

The desired 3D shapes are presented in Figure 15. The cube has a side length of 5 cells, whereas the sphere has a radius of 4 cells. Fitness mean and standard deviation from 100 runs in the NetLogo model for 3D shapes, as well as the evolved number of iterations for the final chromosomes, are presented in Table 4.

Figure 16 shows shapes from some of the best runs obtained using the 3D Margolus model. As in the 2D case, cells outside the desired shape are shown in light gray [15].

Table 4. Fitness mean ($\bar{x}$) and standard deviation (σ) from 100 runs of the CA algorithm for 3D shapes. The evolved number of iterations (*Iter.*) is also presented.

Shape	$\bar{x}$	σ	*Iter.*
Cube	0.9690	0.0165	4
Sphere	0.8579	0.0163	6

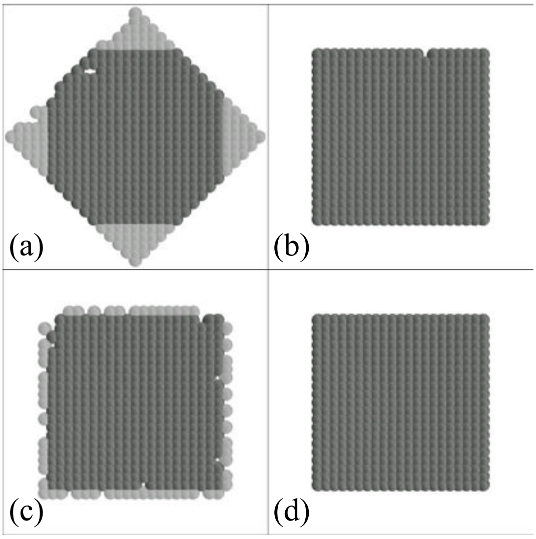


Figure 11. Square shape. Cells outside the desired shape are shown in light gray. Neighborhood description is followed by fitness value. (a) Von Neumann (0.751). (b) Moore (0.998). (c) 2-Radial (0.917). (d) Margolus (1.000).

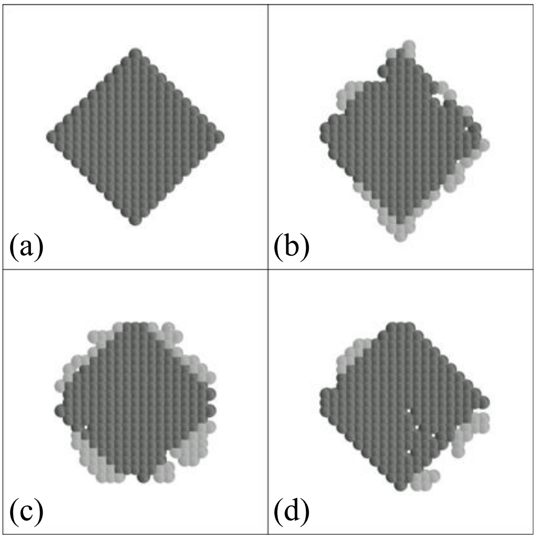


Figure 12. Diamond shape. Cells outside the desired shape are shown in light gray. Neighborhood description is followed by fitness value. (a) Von Neumann (1.000). (b) Moore (0.878). (c) 2-Radial (0.826). (d) Margolus (0.912).

6.3. Chosen neighborhoods for pattern generation

From the above results it was decided to use the Moore and the 3D Margolus neighborhoods for the generation of 2D and 3D cell patterns, respectively, in the simulations that involved the ARNs. Furthermore, in order to reduce simulation times, a smaller $13 \times 13 \times 13$ regular lattice with non-periodic boundaries was used for the pattern generation simulations in 3D.

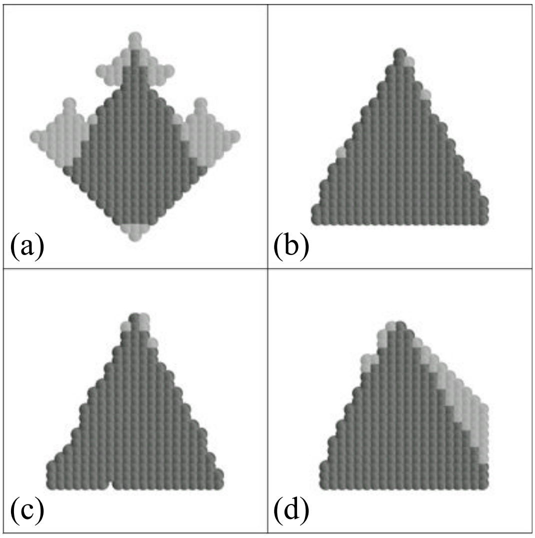


Figure 13. Triangle shape. Cells outside the desired shape are shown in light gray. Neighborhood description is followed by fitness value. (a) Von Neumann (0.580). (b) Moore (0.973). (c) 2-Radial (0.951). (d) Margolus (0.879).

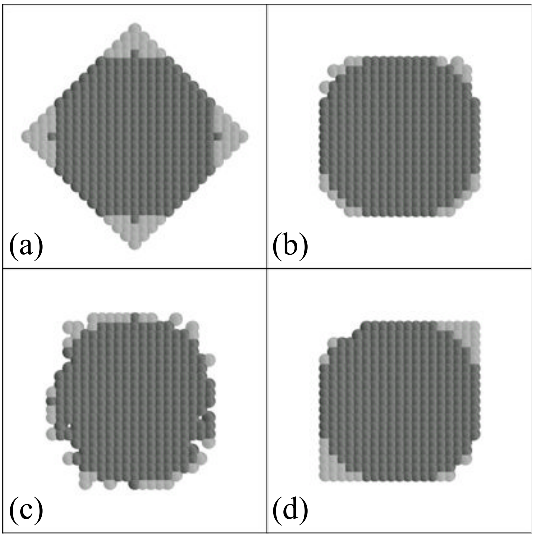


Figure 14. Circle shape. Cells outside the desired shape are shown in light gray. Neighborhood description is followed by fitness value. (a) Von Neumann (0.868). (b) Moore (0.953). (c) 2-Radial (0.901). (d) Margolus (0.939).

Asynchronous updating of cells was originally chosen for the CA implementation, as it had been reported to give more biological-like results [32]. However, in the pattern generation simulations presented in the next section, synchronous updating was used for the sake of reproducibility.

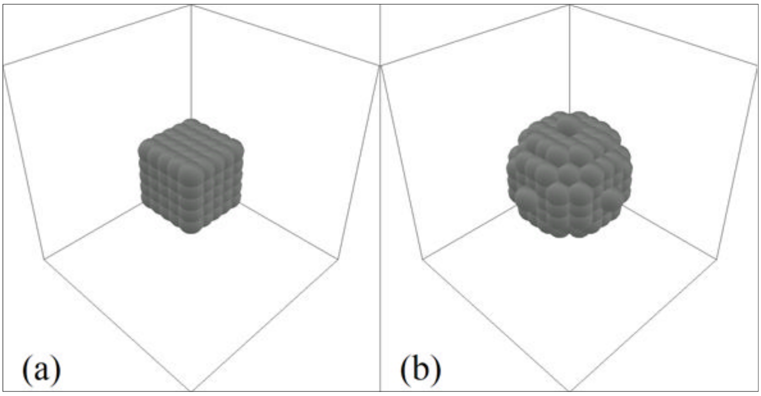


Figure 15. Desired 3D shapes. (a) Cube. (b) Sphere.

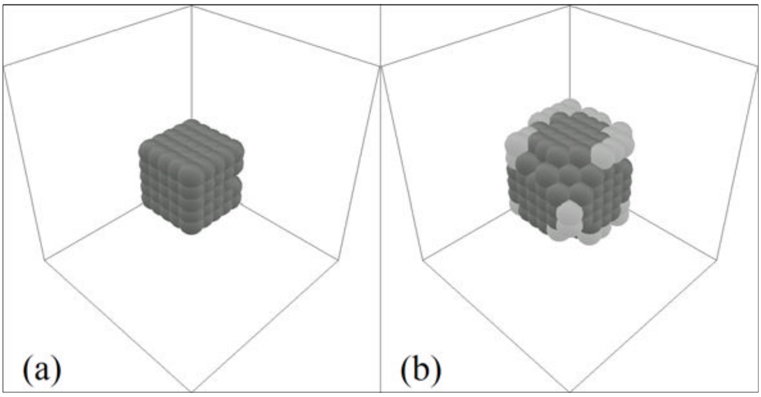


Figure 16. Shapes obtained using the 3D Margolus neighborhood model. Cells outside the desired shape are shown in light gray. Shape description is followed by fitness value. (a) Cube (0.9920), and (b) Sphere (0.8911).

7. Pattern Generation

The GA described in Section 5. was used in all cases to evolve the genome for the desired colored patterns, where each color represented a different structural gene being expressed. The goal was to combine different colored shapes expressed by structural genes in order to obtain a predefined pattern. After a genome was obtained, an initial active cell was placed in the middle of the CA lattice and was allowed to reproduce controlled by the gene activation sequence found by the GA and under the restrictions imposed by the morphogenetic fields. In order to grow the desired pattern with a predefined color and position for each cell, the regulatory genes in the ARN had to evolve to be activated in an appropriate sequence and for a specific number of iterations inside the allowed space defined by the morphogenetic fields. Not all GA experiments produced a genome capable of generating the desired pattern.

The artificial development models were applied to what is known as the *French flag problem*. The problem of generating a French flag pattern was first introduced by Wolpert in the late 1960s when trying to formulate the problem of cell pattern development and

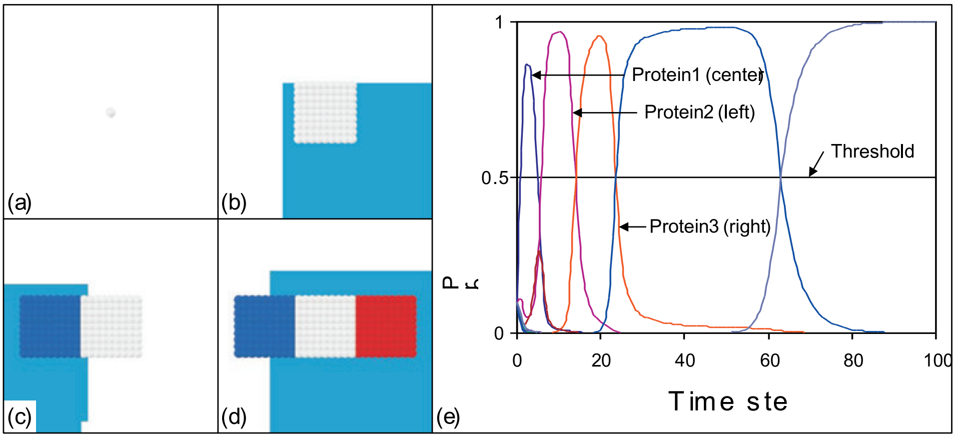


Figure 17. Growth of a 2D French flag pattern. Morphogenetic fields for each gene are shown in light blue. (a) Initial cell. (b) Central white square with morphogenetic field for gene 1 (square). (c) White central square and left blue square with morphogenetic field for gene 2 (extend to left). (d) Finished flag pattern with morphogenetic field for gene 3 (extend to right). (e) Graph of the protein concentration change from the genome expressing the French flag pattern; the unlabeled lines correspond to proteins from regulatory genes that are not associated with structural genes.

regulation in living organisms [33], and it has been used since then by some authors to study the problem of artificial pattern development [34]. In order to grow a French flag pattern in 2D, three different structural genes were first evolved. The first gene drove the creation of the central white square, while the next two genes extended the central square to the left and to the right, expressing the blue and the red color, respectively. The last two structural genes were evolved as described in Section 6. (results not shown) and they do not code specifically for a square; instead they extend a vertical line of cells to the left or to the right for as many time steps as they are activated [35].

Figure 17 shows a 27 × 9 French flag grown from the expression of the three structural genes mentioned above. The graph of the corresponding regulatory protein concentration change over time is shown in 17(e). Starting with an initial white cell (a), a white central square is formed from the expression of gene number 1 (b), the left blue square is then grown (c), followed by the right red square (d). The evolved morphogenetic fields are shown for each of the three structural genes. Since the pattern obtained was exactly as desired, the fitness value assigned to the corresponding genome was 1.

In order to grow a solid 3D French flag pattern, three different structural genes were used. Expression of the first gene creates the white central cube, while the other two genes drive cells to extend the lateral walls to the left and to the right simultaneously, expressing the blue and the red color, respectively. These two last genes were evolved previously and they do not necessarily code for a cube, since they only extend a wall of cells to the left and to the right for as many time steps as they are activated, and when unconstrained, they produce a symmetrical pattern along the *x*-axis. The independent expression of these three genes is shown in Fig. 18.

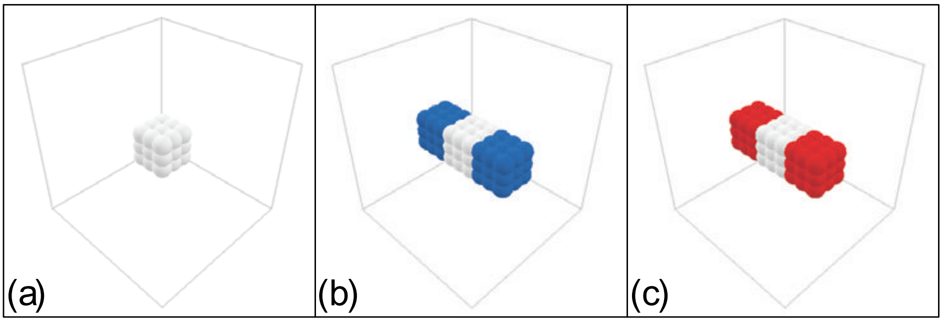


Figure 18. Expression of the three genes used to create a 3D French flag pattern. (a) Create central white cube. (b) Extend blue lateral walls. (c) Extend red lateral walls. The last two genes were activated after the creation of a white central cube

The two genes that extended the lateral walls were activated after a central white cube was first produced. In order to generate the desired French flag pattern, cells expressing one of these two genes should only be allowed to reproduce on each side of the white central cube (left for the blue cube and right for the red cube). This behavior was to be achieved through the use of genomes where the morphogen threshold activation sites evolved to allow growth only in the desired portions of the 3D CA lattice [36].

Figure 19 shows a $9 \times 3 \times 3$ solid French flag pattern grown from the expression of the three structural genes mentioned above. The graph of the corresponding ARN protein concentration change is shown in Fig. 19(e). Starting with an initial white cell (a), a white central cube is formed from the expression of gene number 1 (b), the right red cube is then grown (c), followed by the left blue cube (d). The evolved morphogenetic fields where cells are allowed to grow are depicted in the figure as a translucent volume for each of the three structural genes. Note that for the genes that extend the wall of cells to the sides, the corresponding morphogenetic fields limited growth to the desired direction (red to the right and blue to the left) and produced the desired French flag pattern.

8. Discussion

At the initial stages of the work, the generation of several shapes both convex and non-convex was tried. In geometry, a convex shape has the property that the line segment that joins any two points in the shape is contained within the shape. Given the discrete nature of cell positions in the lattice, convex shapes are harder to define in a CA. We could say that convex shapes in a CA are those where all the filled cells in the shape are circumscribed by the perimeter of a real-valued convex shape with no space left for complete empty cells.

Among the non-convex shapes that were tried, there was a star shape and an L-shaped form. However, it was soon discovered that non-convex shapes are difficult to obtain in a CA with a single rule table and with no provision for allowing filled cells to revert to the empty state. Basically the problem of forming non-convex shapes in a CA with these restrictions is that the same rule table is applied to all cells. Since cells forming a single shape in a CA with a single transition rule are all the same and they only have limited local

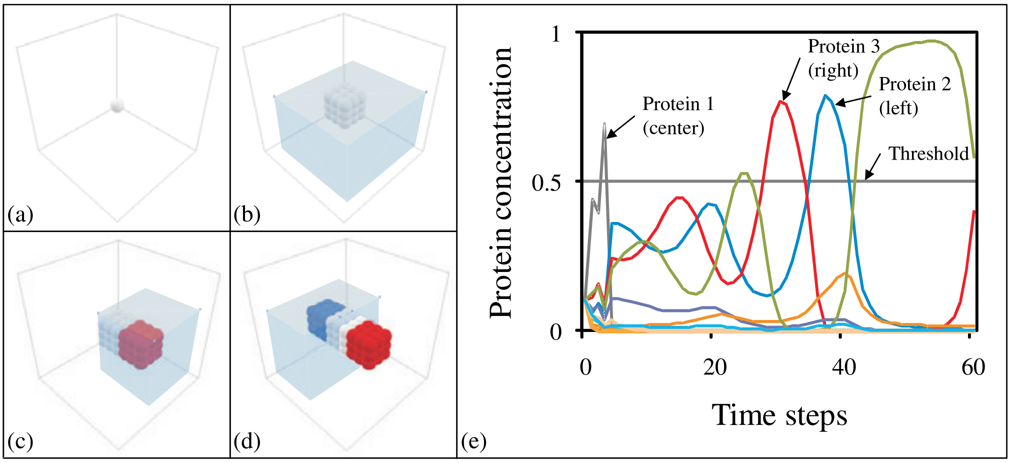


Figure 19. Growth of a 3D French flag pattern. (a) Initial cell. (b) Central white cube with morphogenetic field for gene 1 (cube). (c) Central white cube and right red cube with morphogenetic field for gene 3 (extend red lateral walls). (d) Finished flag pattern with morphogenetic field for gene 2 (extend blue lateral walls). (e) Graph of protein concentration change from the genome expressing the French flag pattern; the unlabeled lines correspond to proteins from regulatory genes that are not associated with structural genes.

information, there is no coordination among cells to differentiate into cells with different roles. Given that cell death and cell displacement are not allowed in the model for the sake of simplicity, once an empty cell becomes occupied or filled, it remains in that state for the rest of the simulation. To compound the problem, a filled cell can only be introduced in the lattice if there is already a filled cell in the neighborhood template.

For the above reasons, all single shapes selected in the end both in 2D and 3D were of the convex type. However, the problem of forming a non-convex pattern in the cellular growth testbed could be solved through the use of cell differentiation, by means of the selective expression of structural genes containing different rule tables.

Results obtained in setting up the framework for single shape generation in 2D showed that although there was not one model that could generate all four shapes with a high degree of accuracy, it was evident that some models were more appropriate than others in building a particular shape. For instance, the von Neumann model was particularly efficient in generating the diamond shape, which is no surprise given the spatial disposition of the neighborhood template itself. However, for the other shapes, the von Neumann model had the worst performance in form generation, possibly due to its “blindness” towards the adjacent diagonally positioned cells.

As for the other neighborhoods, even though the Moore neighborhood template could be viewed as a subset of the 2-Radial neighborhood template, the final chromosomes obtained with the Moore model had higher fitness values for all four shapes. One possible explanation is that the 2-Radial model had a far larger search space than the Moore model, which could make it difficult for the GA to find the fitness maxima. Furthermore, unlike the other neighborhoods where all cells in the outer neighborhood were directly adjacent to the central cell, in the 2-Radial template some cells could be introduced in the lattice without an intermediate cell to be present. In particular the four cells farthest from the neighborhood center could in principle generate a filled cell at the objective cell. This nonetheless gave no apparent advantage to this neighborhood over the other templates.

Although some of the shapes evolved in the cellular growth testbed are not strictly convex, the cells that are not inside the convex shapes are mainly generated by means of the random asynchronous selection of cell reproduction. The shapes shown in the figures of Section 6. are only the product of individual runs. Different runs normally rendered slightly different shapes, and that is the reason for using a probabilistic approach of summarizing the results from 100 runs.

On the average, of the four neighborhoods studied, the most promising models in shape generation were those corresponding to the Moore and the Margolus templates, both with practically the same average fitness (see Table 2). The results obtained with the Margolus template were in most cases comparable to those derived with the Moore neighborhood, even though the former had a chromosome size that was less than a tenth in length than that of the latter. The only case where the Moore neighborhood showed some advantage over the Margolus neighborhood was in the generation of the triangle. Nevertheless, it is possible that the Moore neighborhood would be better at forming more complex shapes than the Margolus template, given its much larger search space. It is for this reason that the Moore neighborhood was chosen as template for the evaluation of the ARN models in the cellular growth testbed. In the 3D case, results showed that the combination of a GA and CA with a 3D Margolus interaction neighborhood was an appropriate choice for modeling 3D shape generation.

As for letting the number of iterations evolve in the cellular growth testbed when generating a predefined shape, this decision turned out to be correct, as the number of iterations needed to create the shapes was not always the same as those intuitively expected. However, in the simulations that best approached the shapes desired, the number of iterations evolved were close to or exactly the same as those expected. It was decided nonetheless that one should not to influence the results with preconceived notions of the expected outcome.

On the other hand, simulations involving the ARN models show that a GA can give reproducible results in evolving an ARN to grow predefined simple 2D and 3D cell patterns starting with a single cell. In particular, it was found that using this ARN model it was feasible to reliably synchronize up to three structural genes to generate a French flag pattern.

One restriction of the ARN models presented is that all cells synchronously follow the same genetic program, as a sort of biological clock. This has obvious advantages for synchronizing the behavior of developing cells, but it would also be desirable that cells had an individual program —possibly a separate ARN— for reacting to local unexpected changes in their environment. Morphogenetic fields provide a means to extract information from the environment, but an independent program would lend more flexibility and robustness to a developing organism. After all, living organisms do contain a series of gene regulatory networks for development and metabolism control. One could even envision either a hierarchy of ARNs, where some ARNs could be used to regulate others ARNs, or a network of ARNs, where all ARNs could influence and regulate each other.

9. Conclusion

The CA-based cellular growth testbed presented proved to be suitable for obtaining simple convex 2D and 3D shapes. Although it was hard to find a CA model that consistently generated convex shapes with a single rule table and no reversion to the empty cell state from the filled cell state, it was found that the expression of different rule tables through the synchronization of an activation sequence in an ARN model could readily do it.

The results presented in this chapter show that a GA can give reproducible results in evolving an ARN to grow predefined simple cellular patterns starting with a single cell. In particular, simulations showed that the combination of a GA and CA with a Moore and a 3D Margolus interaction neighborhood was a feasible choice for modeling pattern generation in 2D and 3D, respectively.

More work is needed to explore pattern formation of more complex forms, both in 2D and 3D. It is also desirable to study pattern formation allowing cell death and cell displacement, as in actual cellular growth. Furthermore, in order to build a more accurate model of the growth process, the use of a more realistic physical environment may be necessary.

One of the long-term goal of this work is to study the emergent properties of the artificial development process. It is conceivable that highly complex structures will one day be built from the interaction of myriads of simpler entities controlled by a development program.

References

[1] Davidson, E.H.: The Regulatory Genome: Gene Regulatory Networks in Development And Evolution. 1st edn. Academic Press (2006)

[2] Eggenberger, P.: Evolving morphologies of simulated 3D organisms based on differential gene expression. In Harvey, I., Husbands, P., eds.: Proceedings of the 4th European Conference on Artificial Life, Springer (1997) 205–213

[3] Reil, T.: Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In: Proceedings of the 5th European Conference on Artificial Life (ECAL), New York, NY, Springer Verlag (1999) 457–466

[4] Banzhaf, W.: Artificial regulatory networks and genetic programming. In Riolo, R.L., Worzel, B., eds.: Genetic Programming Theory and Practice. Kluwer (2003) 43–62

[5] Joachimczak, M., Wróbel, B.: Evo-devo *in silico*: a model of a gene network regulating multicellular development in 3D space with artificial physics. In Bullock, S., Noble, J., Watson, R., Bedau, M.A., eds.: Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems, MIT Press, Cambridge, MA (2008) 297–304

[6] Chavoya, A.: Artificial development. In: Foundations of Computational Intelligence. Volume 1: Learning and Approximation (Studies in Computational Intelligence). Springer Berlin / Heidelberg (2009) 185–215

[7] Bongard, J.: Evolving modular genetic regulatory networks. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002), IEEE Press, Piscataway, NJ (2002) 1872–1877

[8] Kuo, P.D., Leier, A., Banzhaf, W.: Evolving dynamics in an artificial regulatory network model. In Yao, X., Burke, E., Lozano, J., Smith, J., Merelo-Guervós, J., Bullinaria, J., Rowe, J., Tino, P., Kabán, A., Schwefel, H.P., eds.: Proceedings of the Parallel Problem Solving from Nature Conference (PPSN-04). LNCS 3242, Birmingham, UK, Springer, Berlin (September 2004) 571–580

[9] Lindenmayer, A.: Mathematical models for cellular interaction in development, Parts I and II. Journal of Theoretical Biology **18** (1968) 280–315

[10] Meinhardt, H.: Models of Biological Pattern Formation. Academic Press, London (1982)

[11] de Garis, H.: Genetic programming: artificial nervous systems artificial embryos and embryological electronics. In Schwefel, H.P., Männer, R., eds.: Parallel Problem Solving from Nature - Proceedings of 1st Workshop, PPSN 1. Volume 496., Dortmund, Germany, Springer-Verlag, Berlin, Germany (1-3 1991) 117–123

[12] Fleischer, K., Barr, A.H.: A simulation testbed for the study of multicellular development: The multiple mechanisms of morphogenesis. In Langdon, C., ed.: Proceedings of the Workshop on Artificial Life ALIFE'92, Addison-Wesley (1992) 389–416

[13] Kitano, H.: A simple model of neurogenesis and cell differentiation based on evolutionary large-scale chaos. Artificial Life **2**(1) (1994) 79–99

[14] Kumar, S., Bentley, P.J.: On Growth, Form and Computers. Academic Press, London (2003)

[15] Chavoya, A., Duthen, Y.: Using a genetic algorithm to evolve cellular automata for 2D/3D computational development. In: GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation, New York, NY, USA, ACM Press (2006) 231–232

[16] de Garis, H.: Artificial embryology and cellular differentiation. In Bentley, P.J., ed.: Evolutionary Design by Computers. Morgan Kaufmann Publishers, Inc., San Francisco, USA (1999) 281–295

[17] Deutsch, A., Dormann, S.: Cellular Automaton Modeling of Biological Pattern Formation: Characterization, Applications, and Analysis. Birkhäuser, Boston (2005)

[18] Wolfram, S.: Statistical mechanics of cellular automata. Reviews of Modern Physics **55** (1983) 601–644

[19] Mitchell, M., Crutchfield, J.P., Das, R.: Evolving cellular automata with genetic algorithms: A review of recent work. In: Proceedings of the First International Conference on Evolutionary Computation and its Applications (EvCA'96), 1996., Moscow, Russia (1996)

[20] Hordijk, W., Crutchfield, J.P., Mitchell, M.: Mechanisms of emergent computation in cellular automata. In: PPSN V: Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, London, UK, Springer-Verlag (1998) 613–622

[21] Arata, H., Takai, Y., Takai, N.K., Yamamoto, T.: Free-form shape modeling by 3d cellular automata. In: Shape Modeling International (SMI '99), IEEE Computer Society (1999) 242–264

[22] Wu, P., Wu, X., Wainer, G.A.: Applying cell-devs in 3d free-form shape modeling. In: Cellular Automata, 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004. (2004) 81–90

[23] Wilensky, U.: NetLogo. http://ccl.northwestern.edu/netlogo/ (1999) Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.

[24] 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

[25] Carroll, S.B., Grenier, J.K., Weatherbee, S.D.: From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design. 2nd edn. Blackwell Science (2004)

[26] Chavoya, A., Duthen, Y.: A cell pattern generation model based on an extended artificial regulatory network. BioSystems **94**(1) (2008) 95–101

[27] Chavoya, A., Duthen, Y.: Evolving an artificial regulatory network for 2D cell patterning. In: Proceedings of the 2007 IEEE Symposium on Artificial Life (CI-ALife'07), IEEE Computational Intelligence Society (2007) 47–53

[28] Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge, MA, USA (1992)

[29] Mitchell, M.: An introduction to genetic algorithms. MIT Press, Cambridge, MA, USA (1996)

[30] Breukelaar, R., Bäck, T.: Using a genetic algorithm to evolve behavior in multi dimensional cellular automata: emergence of behavior. In: GECCO '05. (2005) 107–114

[31] Chavoya, A., Duthen, Y.: Evolving cellular automata for 2D form generation. In: Proceedings of the Ninth International Conference on Computer Graphics and Artificial Intelligence 3IA'2006. (2006) 129–137

[32] Schonfisch, B., de Roos, A.: Synchronous and asynchronous updating in cellular automata. BioSystems **51** (1999) 123–143

[33] Wolpert, L.: The French flag problem: a contribution to the discussion on pattern development and regulation. In Waddington, C., ed.: Towards a Theoretical Biology. Edinburgh University Press, New York, NY, USA (1968) 125–133

[34] Miller, J.F., Banzhaf, W.: Evolving the program for a cell: from French flags to Boolean circuits. In Kumar, S., Bentley, P.J., eds.: On Growth, Form and Computers. Academic Press (October 2003) 278–301

[35] Chavoya, A., Duthen, Y.: An artificial development model for cell pattern generation. In: ACAL ’07: Proceedings of the 3rd Australian Conference on Artificial Life, Springer (2007) 61–71

[36] Chavoya, A., Andalon-Garcia, I.R., Lopez-Martin, C., Meda-Campaña, M.E.: 3D cell pattern generation in artificial development. In: NICSO ’10: Proceedings of the 4th International Workshop on Nature Inspired Cooperative Strategies for Optimization, Springer (2010) 127–139

In: Cellular Automata
Editor: Thomas M. Li, pp.119-163

ISBN: 978-1-61761-592-4

Chapter 7

STRUCTURAL AND SYMMETRY ANALYSIS OF DISCRETE DYNAMICAL SYSTEMS

Vladimir V. Kornyak[*]
Joint Institute for Nuclear Research
Dubna, Russia

Abstract

To study discrete dynamical systems of different types — deterministic, statistical and quantum — we develop various approaches.We introduce the concept of a system of discrete relations on an abstract simplicial complex and develop algorithms for analysis of compatibility and construction of canonical decompositions of such systems. To illustrate these techniques we describe their application to some cellular automata. Much attention is paid to study symmetries of the systems. In the case of deterministic systems, we reveal some important relations between symmetries and dynamics. We demonstrate that moving soliton-like structures arise inevitably in deterministic dynamical system whose symmetry group splits the set of states into a finite number of group orbits. We develop algorithms and programs exploiting discrete symmetries to study microcanonical ensembles and search phase transitions in mesoscopic lattice models. We propose an approach to quantization of discrete systems based on introduction of gauge connection with values in unitary representations of finite groups — the elements of the connection are interpreted as amplitudes of quantum transitions. We discuss properties of a quantum description of finite systems. In particular, we demonstrate that a finite quantum system can be embedded into a larger classical system. Computer algebra and computational group theory methods were useful tools in our study.

Keywords: discrete relations, cellular automata, symmetries of discrete systems, discrete gauge principle, quantization, computer algebra

MSC 2000: 37A, 37B, 68W, 81R, 81T.

[*] E-mail address: kornyak@jinr.ru

1. Introduction

There are many reasons — physical, mathematical, and conceptual — to study discrete structures. Discrete systems are important in applications — *nanostructures*, for example, by their nature are discrete, not continuous, formations. From a fundamental point of view, there are many philosophical and physical arguments that discreteness better describes physics[1] at small distances than continuity which arises only as approximation or as a logical limit in considering large collections of discrete structures. As a recent development, let us mention much-discussed E. Verlinde's thermodynamic (entropic) derivation [1] of gravity and Newton's law of inertia from G. 't Hooft's *holographic principle* [2]. The holographic principle conjectures that it is possible to describe physical events in a three-dimensional volume fully by a theory on its temporally varying two-dimensional boundary — *holographic screen* — containing *finite* number of discrete degrees of freedom. Entropy of these degrees of freedom, i.e., number of bits N, is proportional to the area A of the screen: $N = \frac{Ac^3}{G\hbar}$.[2] In more speculative sense, the whole universe is a finite two-dimensional information structure on the *cosmological horizon*, and observable three dimensions are only an effective description at macroscopic scales and at low energies. Verlinde shows that the laws of Newton and the Einstein equations come out directly and unavoidably from the holographic principle. The gravity appears to be an *entropic force* arising in systems with many degrees of freedom by the statistical tendency to increase its entropy — like osmosis or elasticity of polymers. Verlinde derived his results combining holography $\left(N = \frac{Ac^3}{G\hbar}\right)$, the equipartition rule (assumption on even distribution of energy over N bits), 1st law of thermodynamics $(dE = TdS - Fdx)$ and several additional standard relations. To introduce thermodynamics, i.e., to construct *canonical partition function*, there is no need to know details of microscopic dynamics. It suffices to know about *energy* and number of states. Of course, the fundamental problem about laws governing bit dynamics on holographic screens remains unsolved. Since Planck scales are experimentally unavailable — the Planck length is about 10^{-35} meters, i.e., far below the spacial resolution of particle accelerators (nowadays about 10^{-18} meters) — the construction and study of various discrete dynamical models is one of the possible approaches.

In this chapter we consider three types of discrete dynamical systems: deterministic, mesoscopic statistical and quantum.

We begin with a general discussion of dyscrete dynamical systems. The most fundamental concepts are a discrete time and a set of states evolving in the time. A space is considered as a derived concept providing the set of states with the specific structure of a set of functions on the points of space with values in some set of local states. We give an illustration of how a space-time may arise in simple models of discrete dynamics. Then we discuss symmetries of space and local states and how these symmetries can be combined into a single group of symmetries of the system as a whole.

We introduce the concept of a system of *discrete relations on an abstract simplicial*

[1] Of course, the question of "whether the real world is discrete or continuous" and even "finite or infinite" is rather *metaphysical*, i.e., neither empirical observations nor logical arguments can validate one of the two adoptions — this is a matter of belief or taste.

[2] In theories with *emergent space* this relation may be used as definition of area: each fundamental bit occupies by definition one unit of area.

complex [3, 4], and explain how any system of discrete relations — subsets of Cartesian products of finite sets — acquires the structure of an abstract simplicial complex. This general concept covers many discrete mathematical structures. In particular, it can be considered as generalization of cellular automata or as a set-theoretical analog of systems of polynomial equations — if all factors of the Cartesian product are sets with the same number of elements and this number is *prime power*, than any relation can be expressed by polynomial equation. We describe algorithms for analysing *compatibility* and constructing *canonical decompositions* of discrete relations. As an illustration, we give results of application of the algorithms to some cellular automata, namely, Conway's automaton *Game of Life* and Wolfram's *elementary cellular automata*. For many of the latter automata the canonical decomposition allows to obtain either general solutions in closed form or important information on their global behavior.

Symmetry is a property of fundamental importance for any mathematical or physical structure. Many real world discrete systems, e.g., carbon nanostructures like graphenes and fullerenes, are highly symmetric formations. Symmetries play essential role in the dynamics of the systems. In this chapter we consider connection between symmetries of discrete dynamical systems on graphs — 1-dimensional simplicial complexes — and their dynamics [5, 6]. In the case of *deterministic dynamical systems*, such as cellular automata, non-trivial connections between the lattice symmetries and dynamics are revealed. In particular, we show that formation of moving soliton-like structures — typical examples are "spaceships" in cellular automata — is a direct result of the existence of non-trivial symmetry.

We developed also algorithms exploiting symmetries for computing microcanonical partition functions and for searching phase transitions in *mesoscopic lattice models*.

We consider a class of discrete dynamical models allowing quantum description [7]. Our approach to quantization consists in introduction of gauge connection with values in unitary representation (not necessarily 1-dimensional) of some group of *internal symmetries* — the elements of the connection are interpreted as amplitudes of quantum transitions. The standard quantization is a special case of this construction — Feynman's path amplitude $\mathrm{e}^{i\int Ldt}$ can be interpreted as parallel transport with values in (1-dimensional) fundamental representation $\mathrm{U}(1)$ of the group of phase transformations. For discrete systems it is natural to take a *finite* group as the *quantizing* group, in this case all manipulations — in contrast to the standard quantization — remain within the framework of constructive discrete mathematics requiring no more than the ring of *algebraic integers* (and sometimes the quotient field of this ring). On the other hand, the standard quantization can be approximated by taking 1-dimensional representations of large enough finite groups.

Any approach to quantization leads ultimately to unitary operators acting on a Hilbert space. We discuss peculiarities of quantum description of finite systems, under the assumption that the operators describing quantum behavior are elements of unitary representations of finite groups. We show that in this case any quantum problem can be embedded into a classical one with a larger space of representation.

Computer algebra and computational group theory [8] methods turned out to be quite useful tools in our study of discrete systems.

2. Discrete Dynamics

Generally, *discrete dynamical system* is a set $\mathcal{S} = \{\mathrm{s}_1, \ldots, \mathrm{s}_{N_\mathcal{S}}\}$ of distinguishable states evolving in *discrete time* $t \in \mathcal{T} \cong \mathbb{Z} = \{\ldots, -1, 0, 1, \ldots\}$, i.e., *evolution* or *history* is an element of the set $\mathcal{E} = \mathcal{S}^\mathcal{T}$. *Dynamics* is determined by some *evolution rule* connecting the current state $s_t \in \mathcal{S}$ of the system with its prehistory s_{t-1}, s_{t-2}, $s_{t-3}, \ldots$ Different types of evolution rules are possible. We shall consider here the following types of discrete dynamics.

- Evolution rule of *deterministic dynamical system* is a *functional relation*. This means that the current state is a function of the prehistory:

$$s_t = F\left(s_{t-1}, s_{t-2}, s_{t-3}, \ldots\right). \tag{1}$$

  Cellular automaton is a typical example of deterministic dynamical system.

- *Statistical lattice model* is a sort of non-deterministic dynamical system. This is a special case of *Markov chain*. In statistical lattice model transition from one state to any other is possible with probability controlled by a Hamiltonian.

- *Quantum system* is another important type of non-deterministic dynamical system. The probabilities of transitions between states are expressed in terms of complex-valued transition amplitudes.

Symmetries play an important — central in the case of quantum systems — role in dynamical systems. So we assume the existence of a non-trivial group $\mathcal{W} = \{\mathrm{w}_1 = \mathbf{1}, \mathrm{w}_2,$ $\ldots,$ $\mathrm{w}_{N_\mathcal{W}}\}$[3] acting on the set of states $\mathcal{S}$: $\mathcal{W} \leq \mathrm{Sym}\left(\mathcal{S}\right)$. Action of the group W splits the set of states $\mathcal{S}$ into *orbits* of different sizes: $\mathcal{S} = \bigsqcup\limits_i O_i$ (disjoint union).

2.1. Discrete Dynamical Models with Space

In applications the set of states $\mathcal{S}$ usually has a special structure of a set of functions on some space.The following constructions form the basis for all types of dynamical systems we consider in this chapter:

1. *Space* is a discrete (basically finite) set of points $\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_{N_\mathrm{X}}\}$ provided with the structure of an abstract regular (k-valent) graph.

2. *Space symmetry group* $\mathrm{G} = \{\mathrm{g}_1 = \mathbf{1}, \mathrm{g}_2, \ldots, \mathrm{g}_{N_\mathrm{G}}\}$ is the graph automorphism group: $\mathrm{G} = \mathrm{Aut}(\mathrm{X}) \leq \mathrm{Sym}\left(\mathrm{X}\right)$. We assume that G acts *transitively* on X.

3. *Local space symmetry group* is defined as the *stabilizer* of a vertex x_i in the space group G: $g \in \mathrm{G}_{\mathrm{loc}} = \mathrm{Stab}_\mathrm{G}\left(\mathrm{x}\right)$ means $\mathrm{x}_i g = \mathrm{x}_i$.[4] Due to the transitivity all such subgroups are isomorphic and we shall denote the isomorphism class by $\mathrm{G}_{\mathrm{loc}}$. This is subgroup of the space symmetry group: $\mathrm{G}_{\mathrm{loc}} \leq \mathrm{G}$.

[3] We denote the identity elements by $\mathbf{1}$ for all groups throughout this chapter.

[4] We write group actions *on the right*. This, more intuitive, convention is adopted in both *GAP* and *Magma* – the most widespread computer algebra systems with advanced facilities for computational group theory.

4. Points $x \in \mathrm{X}$ take values in a finite set $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{N_\Sigma}\}$ of *local states*.

5. *Internal symmetry group* $\Gamma = \{\gamma_1 = \mathbf{1}, \gamma_2, \ldots, \gamma_{N_\Gamma}\}$ is a group $\Gamma \leq \mathrm{Sym}\,(\Sigma)$ acting on the set of local states Σ.

6. *States of the whole system* are functions $\sigma(x) \in \Sigma^{\mathrm{X}} = \mathcal{S}$, and the set of evolutions takes the form $\mathcal{E} = \left(\Sigma^{\mathrm{X}}\right)^{\mathcal{T}} = \Sigma^{\mathrm{X}\times\mathcal{T}}$.

7. We define the *whole symmetry* groups W unifying space G and internal Γ symmetries as equivalence classes of split group extensions of the form

$$\mathbf{1} \to \Gamma^{\mathrm{X}} \to \mathrm{W} \to \mathrm{G} \to \mathbf{1},$$

where Γ^{X} is the set of Γ-valued functions on X. (More detailed description of this construction see in Sect. 2.1.3.)

The separation of the set $\mathcal{S}$ into "space" and "local states" is not fundamental — it is model- and interpretation-dependent. An example of a system with a somewhat non-standard notion of space is a quantum computer. Here the space X is the set of N_{X} qubits, the set of local states Σ is $\{0,1\}$. The whole set of states $\mathcal{S} = \{0,1\}^{\mathrm{X}}$ contains $2^{N_{\mathrm{X}}}$ elements.

2.1.1. Example of Discrete Model with Emergent Space-time.

Modern fundamental theories, in particular the string theory, provide evidence that space is an emergent phenomenon [9], arising from more basic concepts. We demonstrate here that if we have a concept of time then discrete space-time structures may arise under very simple and general assumptions. It is sufficient to have a time-labelled sequence of events and ability to distinguish different types of the events. Then space dimensions arise as the counters of events of different types.

Let us consider a set of states (symbols) $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{N+1}\}$ and assume that it is possible to observe the sequences (histories) $h = s_0, s_1 \cdots s_t$, where $s_i \in \Sigma$. Let us define a space-time point p as equivalence class of sequences with equal numbers of occurrences of each symbol, i.e., p is a "commutative monomial" of the total degree t described by $N+1$ non-negative integers: $p = (n_1, \ldots, n_{N+1})$, $n_1 + \cdots + n_{N+1} = t$, $n_i \in \mathbb{Z}_{\geq 0}$ is multiplicity of symbol σ_i in the history h. The concepts of "causality" and "light cones" arises naturally. The "speed of light limitation" is simply impossibility to get more than t symbols ("perceptions") in t observations — in terms of monomials the "past light cone" is the set of divisors of the monomial p, the "future light cone" is the set of its multiples, see Fig. 1.

The union of all possible histories form a *causal network*. As to modelling continuous Euclidean spaces by this structure, the system of discrete points can be embedded into a continuum in many different ways: as a set of discrete points into a continuous space of arbitrary non-zero dimension[5], as a network into a three-dimensional space[6]. To separate space from the space-time one should introduce a rule identifying points at different

[5] For example, the map $p \longrightarrow \alpha_1 n_1 + \cdots + \alpha_{N+1} n_{N+1} \in \mathbb{R}^1$, where α_i are independent irracionals, provides one-to-one embedding of the set of points into $\mathbb{R}^1$.

[6] A network, as a locally finite 1-dimensional simplicial complex, can always be embedded into $\mathbb{R}^3$.

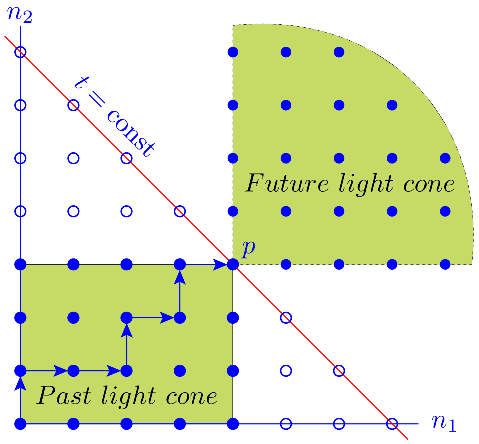
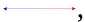
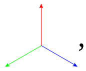
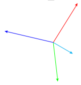
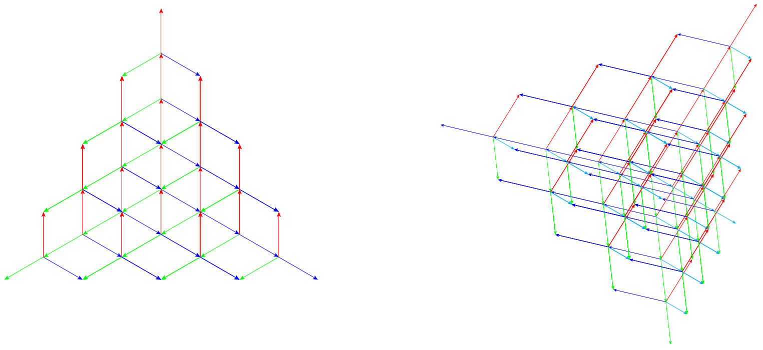


Figure 1. Space-time point p is equivalence class of paths with equal numbers of $n_1, n_2, \ldots, n_{N+1}$.

times. The identification can be any causality-respecting projection onto the set $t = \mathrm{const}$. To construct an illustrative discrete model of this section we use the following projection. Let us identify the symbols σ_i with $N+1$ unit vectors forming regular simplex in an N-dimensional Euclidean space. These systems of vectors (network generating sets) look like , ,  for $N = 1, 2, 3$, respectively. The space lattices generated by these sets in four time steps $(t = 4)$ for the cases $N = 2$ and $N = 3$ are shown in the figure



With these prerequisites, let us construct a simple physical model in 1-dimensional space $(N = 1)$. We have $\Sigma = \{\sigma_1, \sigma_2\} = \{\rightarrow, \leftarrow\}$, $t = n_1 + n_2 \in \mathbb{Z}_{\geq 0}$. Let us add a little physics by imposing the structure of *Bernoulli trials* on the sequences $h = s_0 s_1 \cdots s_t$. Namely, let us introduce probabilities p_1 and p_2 $(p_2 + p_1 = 1)$ for possible outcomes σ_1 and σ_2 of a single trial. The probability of a separate history h is described by the *binomial distribution*

$$P\left(n_1, n_2\right) = \frac{(n_1 + n_2)!}{n_1! n_2!} p_1^{n_1} p_2^{n_2}. \qquad (2)$$

From this model we can see that the behavior of a discrete system may differ essentially from the behavior of its continuous approximation. Applying Stirling's approximation to (2) and introducing new variables $x = n_1 - n_2,\ v = p_1 - p_2$ — let us call them "space" and "velocity", respectively — we obtain

$$P\left(x,t\right) \approx \widetilde{P}\left(x,t\right) = \frac{1}{\sqrt{1-v^2}}\sqrt{\frac{2}{\pi t}}\exp\left\{-\frac{1}{2t}\left(\frac{x-vt}{\sqrt{1-v^2}}\right)^2\right\}. \tag{3}$$

This is the *fundamental solution* of the *heat* (also known as *diffusion* or *Fokker–Planck*)[7] *equation*:

$$\frac{\partial\widetilde{P}\left(x,t\right)}{\partial t} + v\frac{\partial\widetilde{P}\left(x,t\right)}{\partial x} = \frac{\left(1-v^2\right)}{2}\frac{\partial^2\widetilde{P}\left(x,t\right)}{\partial x^2}. \tag{4}$$

Note that expression (3) — due to the velocity limits $-1 \leq v \leq 1$ in our model — contains "relativistic" fragment $\frac{x-vt}{\sqrt{1-v^2}}$. Note also that at $|v| = 1$ equation (4) is reduced to the *wave equation*

$$\frac{\partial\widetilde{P}\left(x,t\right)}{\partial t} \pm \frac{\partial\widetilde{P}\left(x,t\right)}{\partial x} = 0. \tag{5}$$

Now let us set a problem as is typical in mechanics: find extremal trajectories connecting two fixed points $(0,0)$ and (X,T). As a version of the "least action principle", we adopt here the search of trajectories of maximum probability. The probability of trajectory connecting the points $(0,0)$ and (X,T) and passing through some intermediate point (x,t) is the following *conditional probability*

$$\begin{aligned} P_{(0,0)\to(x,t)\to(X,T)} &= \frac{P(x,t)P(X-x,T-t)}{P(X,T)} \\ &= \frac{t!(T-t)!\left(\frac{T-X}{2}\right)!\left(\frac{T+X}{2}\right)!}{\left(\frac{t-x}{2}\right)!\left(\frac{t+x}{2}\right)!\left(\frac{T-t}{2}-\frac{X-x}{2}\right)!\left(\frac{T-t}{2}+\frac{X-x}{2}\right)!T!}. \end{aligned} \tag{6}$$

The conditional probability computed for approximation (3) takes the form

$$\widetilde{P}_{(0,0)\to(x,t)\to(X,T)} = \frac{T}{\sqrt{\frac{\pi}{2}(1-v^2)tT(T-t)}}\exp\left\{-\frac{\left(Xt-xT\right)^2}{2(1-v^2)tT(T-t)}\right\}. \tag{7}$$

One can see essential differences between (6) and (7):

- exact probabilities (6) do *not depend* on the velocity v (or, equivalently, on the probabilities $p_1,\ p_2$ of a single trial), whereas (7) contains *artificial dependence,*

- it is easy to check that expression (6) allows *many* trajectories with the *same maximum probability*, whereas extremals of (7) are *deterministic trajectories*, namely, straight lines $x = \frac{X}{T}t$. This is a typical example of emergence of deterministic behaviour as a result of the law of large numbers approximation.

[7] The name of the equation depends on interpretation of the function $\widetilde{P}\left(x,t\right)$.

2.1.2. Space Symmetries in More Detail.

A space X in our models has the structure of a graph. Graphs — we shall call them also *lattices* — are sufficient for all our purposes. In particular, they are adequate to introduce gauge and quantum structures. The symmetry group of the space X is the graph automorphism group $\mathrm{G} = \mathrm{Aut}\,(\mathrm{X})$. The *automorphism group* of a graph with n vertices may have up to $n!$ elements. Nevertheless, the most efficient currently algorithm designed by B. McKay [10] determines the graph automorphisms by constructing compact set (no more than $n-1$ elements, but usually much less) of generators of the group.

Very often dynamics of a model is expressed in terms of rules defined on the neighborhoods of lattice vertices. For this sort of models with locally defined evolution rules — typical examples are cellular automata and the Ising model — the above mentioned group of local symmetries $\mathrm{G}_{\mathrm{loc}}$ is essential. Local rules are defined on *orbits* of $\mathrm{G}_{\mathrm{loc}}$ on *edges* from the *neighborhoods* of points x. Fig. 2 shows the symmetry groups G and $\mathrm{G}_{\mathrm{loc}} \leq \mathrm{G}$ for some carbon and hydrocarbon molecules.

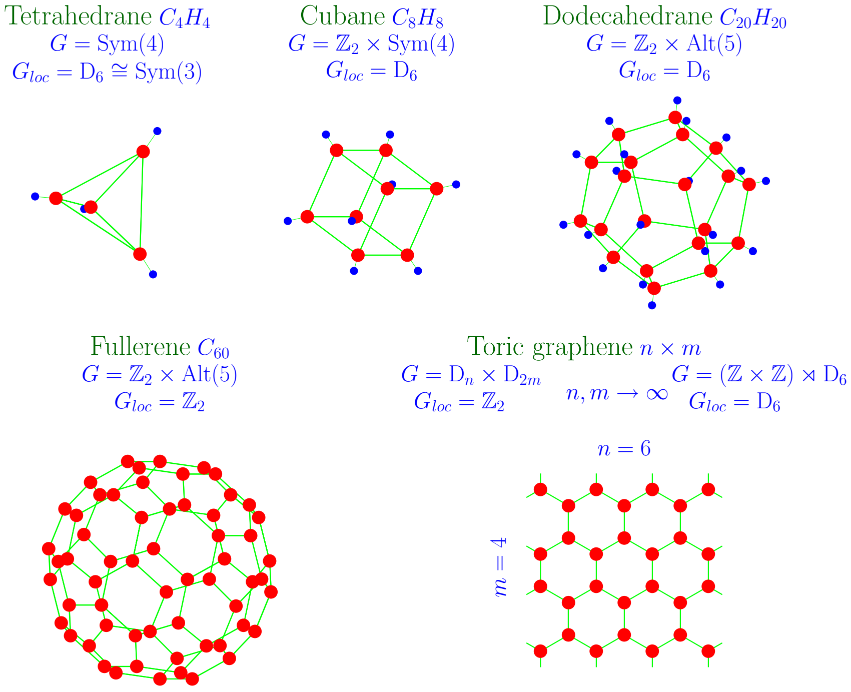
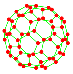
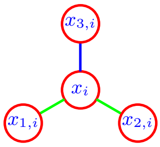

Tetrahedrane C_4H_4
$G = \mathrm{Sym}(4)$
$G_{loc} = \mathrm{D}_6 \cong \mathrm{Sym}(3)$
Cubane C_8H_8
$G = \mathbb{Z}_2 \times \mathrm{Sym}(4)$
$G_{loc} = \mathrm{D}_6$
Dodecahedrane $C_{20}H_{20}$
$G = \mathbb{Z}_2 \times \mathrm{Alt}(5)$
$G_{loc} = \mathrm{D}_6$
Fullerene C_{60}
$G = \mathbb{Z}_2 \times \mathrm{Alt}(5)$
$G_{loc} = \mathbb{Z}_2$
Toric graphene $n \times m$
$G = \mathrm{D}_n \times \mathrm{D}_{2m}$
$G_{loc} = \mathbb{Z}_2$
$n, m \to \infty$
$G = (\mathbb{Z} \times \mathbb{Z}) \rtimes \mathrm{D}_6$
$G_{loc} = \mathrm{D}_6$
$n = 6$
$m = 4$

Figure 2. Symmetries of 3-valent (hydro)carbon nanostructures.

Let us consider the role of the local group $\mathrm{G}_{\mathrm{loc}}$ in more detail using the *buckyball*  as an example. The incarnations of this 3-valent graph include in particular:

- the *Caley graph of the icosahedral group*[8] $\mathrm{Alt}(5)$ (in mathematics);
- the *molecule of fullerene* C_{60} (in carbon chemistry).

[8] The classical book by F. Klein [11] is devoted entirely to this group.

The symmetry group of the buckyball is $\mathrm{G} = \mathrm{Aut}\left(\mathrm{X}\right) = \mathbb{Z}_2 \times \mathrm{Alt}(5)$. The neighborhood of a vertex x_i takes the form 

. The stabilizer of x_i is $\mathrm{G}_{\mathrm{loc}} = \mathrm{Stab}_{\mathrm{G}}\left(x_i\right) = \mathbb{Z}_2$. The set of neighborhood edges contains three elements:

$$E_i = \left\{e_{1,i} = \left(x_i, x_{1,i}\right), e_{2,i} = \left(x_i, x_{2,i}\right), e_{3,i} = \left(x_i, x_{3,i}\right)\right\}.$$

The set of orbits of $\mathrm{G}_{\mathrm{loc}}$ on E_i consists of two orbits:

$$\Omega_i = \left\{\omega_{1,i} = \left\{e_{1,i}, e_{2,i}\right\}, \omega_{2,i} = \left\{e_{3,i}\right\}\right\},$$

i.e., the stabilizer does not move the edge $\left(x_i, x_{3,i}\right)$ and swaps $\left(x_i, x_{1,i}\right)$ and $\left(x_i, x_{2,i}\right)$. This asymmetry results from different roles the edges play in the structure of the buckyball: $\left(x_i, x_{1,i}\right)$ and $\left(x_i, x_{2,i}\right)$ are edges of a pentagon adjacent to x_i , whereas $\left(x_i, x_{3,i}\right)$ separates two hexagons; in the carbon molecule C_{60} the edge $\left(x_i, x_{3,i}\right)$ corresponds to the double bond, whereas others are the single bonds.

Naturally formulated local rules determining behavior of a system must respect decompositions of neighborhoods into the orbits of the group of local symmetries. For example, the Hamiltonian of the Ising model on the buckyball must depend on two, generally different, coupling constants J_{12} and J_3. Moreover, the coupling constants may be of different types — ferromagnetic or antiferromagnetic — and this may lead to interesting behavior of the model. Such natural Hamiltonian should take the form

$$H_{bucky} = -\frac{1}{2}\sum_i s_i\left[J_{12}\left(s_{1,i} + s_{2,i}\right) + J_3 s_{3,i}\right] - B\sum_i s_i\,, \qquad (8)$$

where $s_i, s_{1,i}, s_{2,i}, s_{3,i} \in \Sigma = \{-1, 1\}$. In a similar way the local rule for a cellular automaton on the buckyball must have the form

$$x_i' = f\left(x_i, x_{1,i}, x_{2,i}, x_{3,i}\right)\ ,$$

where function f must be symmetric with respect to variables $x_{1,i}$ and $x_{2,i}$, i.e.,

$$f\left(x_i, x_{1,i}, x_{2,i}, x_{3,i}\right) \equiv f\left(x_i, x_{2,i}, x_{1,i}, x_{3,i}\right).$$

2.1.3. Unification of Space and Internal Symmetries.

Having the groups G and Γ acting on X and Σ, respectively, we can combine them into a single group $\mathrm{W} \leq \mathrm{Sym}\left(\Sigma^{\mathrm{X}}\right)$ which acts on the states $\mathcal{S} = \Sigma^{\mathrm{X}}$ of the whole system. The group W can be identified, as a set, with the *Cartesian product* $\Gamma^{\mathrm{X}} \otimes \mathrm{G}$, where Γ^{X} is the set of Γ-valued functions on X. That is, every element $u \in \mathrm{W}$ can be represented in the form $u = \left(\alpha(x),\ a\right)$, where $\alpha(x) \in \Gamma^{\mathrm{X}}$ and $a \in \mathrm{G}$. *A priori* there are different possible ways to combine G and Γ into a single group. So selection of possible combinations should be guided by some natural (physical) reasons. General arguments convince that the required combination W should be a *split extension* of the group G by the group Γ^{X}. In physics, it is usually assumed that the space and internal symmetries are independent, i.e., W is the *direct product* $\Gamma^{\mathrm{X}} \times \mathrm{G}$ with action on Σ^{X} and multiplication rules:

$$\begin{aligned} \sigma(x)\,(\alpha\,(x),\;a) &= \sigma\,(x)\,\alpha\,(x) && \textit{action}, \\ (\alpha\,(x),\;a)\,(\beta\,(x),\;b) &= (\alpha\,(x)\,\beta\,(x),\;ab) && \textit{multiplication}. \end{aligned} \tag{9}$$

Another standard construction is the *wreath product* $\Gamma \wr_{\mathrm{X}} \mathrm{G}$ having a structure of the semidirect product $\Gamma^{\mathrm{X}} \rtimes \mathrm{G}$ with action and multiplication

$$\begin{aligned} \sigma(x)\,(\alpha\,(x),\;a) &= \sigma\left(xa^{-1}\right)\alpha\left(xa^{-1}\right), \\ (\alpha\,(x),\;a)\,(\beta\,(x),\;b) &= (\alpha\,(x)\,\beta\,(xa),\;ab)\,. \end{aligned} \tag{10}$$

These examples are generalized by the following
Statement. *There are equivalence classes of split group extensions*

$$1 \to \Gamma^{\mathrm{X}} \to \mathrm{W} \to \mathrm{G} \to 1 \tag{11}$$

determined by antihomomorphisms[9] $\mu : \mathrm{G} \to \mathrm{G}$. *The equivalence is described by arbitrary function* $\kappa : \mathrm{G} \to \mathrm{G}$. *The explicit formulas for main group operations —* action *on* Σ^{X}*,* multiplication *and* inversion *— are*

$$\sigma(x)\,(\alpha\,(x),\;a) = \sigma\,(x\mu(a))\,\alpha\,(x\kappa(a))\,, \tag{12}$$

$$(\alpha\,(x),\;a)\,(\beta\,(x),\;b) = \left(\alpha\left(x\kappa(ab)^{-1}\mu(b)\kappa(a)\right)\beta\left(x\kappa(ab)^{-1}\kappa(b)\right),\;ab\right), \tag{13}$$

$$(\alpha(x),\;a)^{-1} = \left(\alpha\left(x\kappa\left(a^{-1}\right)^{-1}\mu(a)^{-1}\kappa(a)\right)^{-1},\;a^{-1}\right). \tag{14}$$

This statement follows from the general description of the structure of split extensions of a group G by a group H: all such extensions are determined by the homomorphisms from G to $\mathrm{Aut}\,(H)$ (see, e.g., [12]). Specializing this description to the case when H is the set of Γ-valued function on X and G acts on arguments of these functions we obtain our statement. The *equivalence* of extensions with the same antihomomorfism μ but with different functions κ is expressed by the commutative diagram

$$\begin{tikzcd} \mathbf{1} \arrow[r] & \Gamma^{\mathrm{X}} \arrow[r] \arrow[d, equal] & \mathrm{W} \arrow[r] \arrow[d, "K"'] & \mathrm{G} \arrow[r] \arrow[d, equal] & \mathbf{1} \\ \mathbf{1} \arrow[r] & \Gamma^{\mathrm{X}} \arrow[r] & \mathrm{W}' \arrow[r] & \mathrm{G} \arrow[r] & \mathbf{1} \end{tikzcd}\;, \tag{15}$$

where the mapping K takes the form $K : (\alpha(x),\;a) \mapsto (\alpha\,(x\kappa(a)),\;a)\,.$

Note that the standard *direct* and *wreath* products are obtained from this general construction by choosing antihomomorphisms $\mu(a) = 1$ and $\mu(a) = a^{-1}$, respectively. As to the *arbitrary* function κ, the choices $\kappa(a) = 1$ and $\kappa(a) = a^{-1}$, respectively, are generally used in the literature.

[9] The term ‘antihomomorphism’ means that $\mu(a)\mu(b) = \mu(ba)$.

In our computer programs (written in C) the group W is specified by two groups G and Γ and two functions $\mu(a)$ and $\kappa(a)$ implemented as arrays. It is convenient in computations to use the following specialization: $\mu(a) = a^{-m}$ and $\kappa(a) = a^k$. For such a choice formulas (12)-14 take the form

$$\sigma(x)\left(\alpha\left(x\right),\ a\right) = \sigma\left(xa^{-m}\right)\alpha\left(xa^{k}\right), \tag{16}$$

$$\left(\alpha\left(x\right),\ a\right)\left(\beta\left(x\right),\ b\right) = \left(\alpha\left(x(ab)^{-k-m}a^{k+m}\right)\beta\left(x(ab)^{-k}b^{k}\right),\ ab\right), \tag{17}$$

$$\left(\alpha(x),\ a\right)^{-1} = \left(\alpha\left(xa^{2k+m}\right)^{-1},\ a^{-1}\right). \tag{18}$$

Here k is *arbitrary* integer, but m is restricted only to two values: $m = 0$ and $m = 1$, i.e., such specialization does not cover other than, respectively, *direct* and *wreath* types of split extentions. On the other hand, the antihomomorphisms $\mu(a) = 1$ and $\mu(a) = a^{-1}$ exist for any group, while others depend on the particular structure of a group. Note that actions of G on any function $f(x)$ are called *trivial* and *natural* for $\mu(a) = 1$ and $\mu(a) = a^{-1}$, respectively.

3. Structural Analysis of Discrete Relations

The methods of compatibility analysis, such as the Gröbner basis computation or reduction to involutive form, are widely used to study systems of polynomial and differential equations. In this section we develop similar techniques for discrete systems, in particular, for cellular automata.

Let us consider the Cartesian product $\Sigma^n = \Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_n$, i.e., the set of ordered n-tuples $(\sigma_1, \sigma_2, \ldots, \sigma_n)$, with $\sigma_i \in \Sigma_i$ for each i. By definition, n-*ary relation* is any subset of the n-dimensional hyperparallelepiped Σ^n. We assume that Σ_i are finite sets of $q_i = |\Sigma_i|$ elements that we shall call *states*.

We can treat n dimensions of the hyperparallelepiped Σ^n as elements of a set of points $X = \{x_1, x_2, \ldots, x_n\}$. To make this initially amorphous set into a "space" (or "space-time") we should provide X with a structure determining how "close" to each other are different points. The relevant mathematical abstraction of such a structure is an abstract simplicial complex. The natural concept of space assumes the homogeneity of its points. This means that there exists a symmetry group acting transitively on X, i.e., providing possibility to "move" any point into any other. The homogeneity is possible only if all Σ_i are equivalent. Let us denote the equivalence class by Σ. We can represent Σ canonically in the form $\Sigma = \{0, \ldots, q-1\}$, $q = |\Sigma|$.

If the number of states is a prime power, $q = p^m$, we can additionally equip the set Σ with the structure of the Galois field $\mathbb{F}_q$. Using the functional completeness of polynomials — this means that *any* function can be represented as polynomial — over finite fields [13], we can represent any k-ary relation on Σ as a set of zeros of some polynomial belonging to the ring $\mathbb{F}_q\left[x_1, \ldots, x_k\right]$. Thus, the set of relations can be regarded as a system of polynomial equations. Although this description is not necessary (and does not work, if Σ_i are different sets or q is not prime power), it is useful due to our habit to employ polynomials wherever possible and capability of applying different advanced tools of polynomial algebra, such as, for example, the Gröbner bases.

An *abstract simplicial complex* (see, e.g., [14]) $K = (X, \Delta)$ is determined by a set of points $X = \{x_1, x_2, \ldots, x_n\}$ and an assembly Δ of subsets of X, which are called *simplices,* such that (a) for all $x_i \in X$ $\{x_i\} \in \Delta$ and (b) if $\tau \subseteq \delta \in \Delta$, then $\tau \in \Delta$. The subsets of a simplex — they are also simplices due to (b) — are called *faces*. Condition (a) means that all one-element subsets are simplices. Clearly, the structure of the complex K, i.e., the set Δ, is uniquely determined by the simplices that are *maximal* by inclusion. *Dimension* of a simplex δ is the number $\dim \delta = |\delta| - 1$. This definition is motivated by the fact that $k + 1$ points immersed in the general position into the Euclidean space $\mathbb{R}^{n \geq k}$ form a k-dimensional convex polyhedron. The dimension of a complex K is defined as the maximum dimension of all simplices in K: $\dim K = \max\limits_{\delta \in \Delta} \dim \delta$. From the point of view of abstract combinatorial topology, no matter how the complex can be immersed into the space $\mathbb{R}^n$ — it is essential only how its simplices are connected with each other. However, it follows from the Nöbeling–Pontryagin theorem that any (locally finite) abstract k-dimensional complex can be geometrically realized in the space $\mathbb{R}^{2k+1}$. We will show below that, for any n-ary relation $R \subseteq \Sigma^n$, one can regularly and uniquely construct some abstract simplicial complex.

3.1. Basic Definitions and Constructions

In addition to k-simplices, which are singled out sets of $k + 1$ points, we need to consider arbitrary sets of point. For brevity, we shall call sets containing k points by k-sets. Dealing with systems of relations defined on different sets of points, it is necessary to establish correspondence between the points and dimensions of the hypercube Σ^k. This is achieved by using exponential notation. The notation $\Sigma^{\{x_i\}}$ fixes Σ as the set of values of the point x_i. For the k-set $\delta = \{x_1, \ldots, x_k\}$, we introduce the notation $\Sigma^\delta = \Sigma^{\{x_1\}} \times \cdots \times \Sigma^{\{x_k\}}$. The set δ is called the *domain* of the relation R^δ. We will call the whole hypercube Σ^δ a *trivial* relation. Accordingly, $R^\delta \subseteq \Sigma^\delta$ denotes a relation given on the set of points δ.

3.1.1. Relations

Thus, we have:

Definition 1 (relation). A *relation* R^δ on the set of points $\delta = \{x_1, \ldots, x_k\}$ is any subset of the hypercube Σ^δ; i.e., $R^\delta \subseteq \Sigma^\delta$.
The relation R^δ can be regarded as the Boolean-valued function $R^\delta : \Sigma^\delta \rightarrow \{0, 1\}$. We can think of x_i's as variables taking values in Σ and write the relation as

$$a = R^\delta\left(x_1, \ldots, x_k\right), \;\; a \in \{0, 1\}.$$

An important special case of relations:

Definition 2 (functional relation). A *relation* R^δ on the set of points
$\delta = \{x_1, \ldots, x_k\}$ is called *functional* if there is a position $i \in (1, \ldots, k)$ such that
for any $\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_k, \varsigma, \tau \in \Sigma$
from $(\sigma_1, \ldots, \sigma_{i-1}, \varsigma, \sigma_{i+1}, \ldots, \sigma_k) \in R^\delta$ and $(\sigma_1, \ldots, \sigma_{i-1}, \tau, \sigma_{i+1}, \ldots, \sigma_k) \in R^\delta$

it follows that $\varsigma = \tau$.
In terms of variables the functional relation R^δ can be written in the form

$$x_i = F\left(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k\right), \text{ where } F : \Sigma^{\delta\setminus\{x_i\}} \to \Sigma.$$

We need to be able to extend relations from subsets of points to larger sets:

Definition 3 (extension of relation). For given set of points δ, its subset $\tau \subseteq \delta$ and relation R^τ on the subset τ, we define the *extension* of R^τ as the relation

$$R^\delta = R^\tau \times \Sigma^{\delta\setminus\tau}.$$

This definition, in particular, allows the relations $R^{\delta_1}, \ldots, R^{\delta_m}$ defined on different domains to be extended to the common domain, i.e., to the union $\delta_1 \cup \cdots \cup \delta_m$.

Logical implications of the relations are defined in a natural way:

Definition 4 (consequence of relation). A relation Q^δ is called a *consequence* of the relation R^δ if $R^\delta \subseteq Q^\delta \subseteq \Sigma^\delta$; i.e., Q^δ is arbitrary *superset* of the set R^δ.

The relation R^δ may have many different consequences: their total number (including R^δ itself and the trivial relation Σ^δ) is evidently equal to $2^{\left|\Sigma^\delta\right|-\left|R^\delta\right|}$.

It is natural to single out the consequences that can be reduced to relations on smaller sets of points:

Definition 5 (proper consequence). A *nontrivial* relation Q^τ is called the *proper consequence* of the relation R^δ if τ is a *proper* subset of δ (i.e., $\tau \subset \delta$) and the relation $Q^\tau \times \Sigma^{\delta\setminus\tau}$ is a consequence of R^δ.

We call relations that have no proper consequences the *prime relations*.

3.1.2. Compatibility of Systems of Relations

The compatibility of a system of relations can naturally be defined by the intersection of their extensions to the common domain:

Definition 6 (base relation). The *base relation* of the system of relations $R^{\delta_1}, \ldots, R^{\delta_m}$ is the relation

$$R^\delta = \bigcap_{i=1}^{m} R^{\delta_i} \times \Sigma^{\delta_i\setminus\delta}, \text{ where } \delta = \bigcup_{i=1}^{m} \delta_i.$$

Let us make two comments for the polynomial case $q = p^n$, where the standard tool for the compatibility analysis is the Gröbner basis method:

- The compatibility condition determined by the *base relation* can be represented by a *single* polynomial, unlike the Gröbner basis, which is normally a system of polynomials.

- Any possible Gröbner basis of polynomials representing the relations $R^{\delta_1}, \ldots, R^{\delta_m}$ corresponds to some combination of consequences of the *base relation*.

3.1.3. Decomposition of Relations

If a relation has proper consequences, we can try to express it as far as possible in terms of these consequences, i.e., relations on smaller sets of points. To this end we introduce

Definition 7 (canonical decomposition). The *canonical decomposition* of a relation R^δ with proper consequences $Q^{\delta_1}, \ldots, Q^{\delta_m}$ is the relation

$$R^\delta = P^\delta \bigcap \left(\bigcap_{i=1}^{m} Q^{\delta_i} \times \Sigma^{\delta \setminus \delta_i} \right), \tag{19}$$

where the factor P^δ is defined by the following

Definition 8 (principal factor). The *principal factor* of the relation R^δ with proper consequences $Q^{\delta_1}, \ldots, Q^{\delta_m}$ is the relation

$$P^\delta = R^\delta \bigcup \left(\Sigma^\delta \setminus \bigcap_{i=1}^{m} Q^{\delta_i} \times \Sigma^{\delta \setminus \delta_i} \right).$$

The principal factor is the maximally "free" — i.e., the closest to the trivial — relation that, together with the proper consequences, makes it possible to recover the initial relation.

If the principal factor in the canonical decomposition is trivial, the relation is completely reduced to relations on smaller sets of points.

Definition 9 (reducible relation). A relation R^δ is said to be *reducible* if it can be represented as

$$R^\delta = \bigcap_{i=1}^{m} Q^{\delta_i} \times \Sigma^{\delta \setminus \delta_i}, \tag{20}$$

where δ_i are *proper subsets* of δ.

This definition makes it possible to impose a "topology" — i.e., the structure of an abstract simplicial complex with the corresponding theories of homologies, cohomologies, etc. — on an *arbitrary* n-ary relation $R \subseteq \Sigma^n$. This is achieved by

- naming the dimensions of the hypercube Σ^n as the "points" $x_1, \ldots, x_n \in X$,
- decomposing R (which can now be denoted by R^X) into *irreducible* components,
- and defining the *maximal simplices* of the set Δ as the *domains* of irreducible components of the relation R^X.

3.1.4. On Representation of Relations in Computer

A few words are needed about computer implementation of relations. To specify a k-ary relation R^k we should mark its points within the k-dimensional hypercube (or hyperparallelepiped) Σ^k, i.e., define a *characteristic function* $\chi : \Sigma^k \to \{0, 1\}$, with $\chi(\vec{\sigma}) = 1$ or 0 according as $\vec{\sigma} \in R^k$ or $\vec{\sigma} \notin R^k$. Here $\vec{\sigma} = (\sigma_0, \sigma_1, \ldots, \sigma_{k-1})$ is a point of the hypercube. The simplest way to implement the characteristic function is to enumerate all the q^k hypercube points in some standard, e.g., lexicographic order. Then the relation can be represented by a string of q^k bits $\alpha_0 \alpha_1 \cdots \alpha_{q^k-1}$ in accordance with the table:

σ_0	σ_1	$\ldots$	σ_{k-2}	σ_{k-1}	$i_{\vec{\sigma}}$	$\chi(\vec{\sigma})$
0	0	$\ldots$	0	0	0	α_0
1	0	$\ldots$	0	0	1	α_1
$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$q-2$	$q-1$	$\ldots$	$q-1$	$q-1$	q^k-2	α_{q^k-2}
$q-1$	$q-1$	$\ldots$	$q-1$	$q-1$	q^k-1	α_{q^k-1}

We call this string *bit table* of relation. Symbolically $\mathrm{BitTable}\,[\,i_{\vec{\sigma}}] := \left(\vec{\sigma} \in R^k\right)$. Note that $\vec{\sigma}$ is the ("little-endian") representation of the number $i_{\vec{\sigma}}$ in the radix q:

$$i_{\vec{\sigma}} = \sigma_0 + \sigma_1 q + \cdots + \sigma_i q^i + \cdots + \sigma_{k-1} q^{k-1}.$$

In the case of hyperparallelepiped $\Sigma^k = \Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_k$ one should use the *multi-radix representation* of integers:

$$i_{\vec{\sigma}} = \sigma_0 + \sigma_1 \times q_1 + \cdots + \sigma_i \times q_1 q_2 \cdots q_i + \cdots + \sigma_{k-1} \times q_1 q_2 \cdots q_{k-1},$$

where $0 \le \sigma_i < q_{i+1},\ \ i \in [0, \ldots, k-1]$.

The characteristic function (bit table) can be represented as the *binary* integer

$$\chi = \alpha_0 + \alpha_1 2 + \cdots + \alpha_i 2^i + \cdots + \alpha_{q^k-1} 2^{q^k-1}. \tag{21}$$

Most manipulations with relations are reduced to very efficient bitwise computer commands. Of course, symmetric or sparse (or, vice versa, dense) relations can be represented in a more economical way, but these are technical details of implementation.

3.2. Illustration: Application to Some Cellular Automata

3.2.1. J. Conway's Game of Life

The "Life family" is a set of 2-dimensional, binary (i.e., $\Sigma = \{0, 1\}$; $q = 2$) cellular automata similar to ***Conway's Life***, which rule is defined on 9-cell (3×3) Moore neighborhood and is described as follows. A cell is "born" if it has exactly 3 "alive" neighbors, "survives" if it has 2 or 3 such neighbors, and "dies" otherwise. This rule is symbolized in terms of the "birth"/"survival" lists as B3/S23. Another examples of automata from this family are ***HighLife*** (the rule B36/S23), and ***Day&Night*** (the rule B3678/S34678). The site [15] contains collection of more than twenty rules from the Life family with Java applet to run these rules and descriptions of their behavior.

Generalizing this type of local rules, we define a *k-valent Life rule* as a *binary* rule on a k-valent neighborhood (we adopt that $x_1, \ldots, x_k$ are neighbors of x_{k+1} of the central cell x_{k+1}) described by two *arbitrary* subsets of the set $\{0, 1, \ldots, k\}$. These subsets $B, S \subseteq \{0, 1, \ldots, k\}$ contain conditions for the one-time-step transitions $x_{k+1} \rightarrow x'_{k+1}$ of the forms $0 \rightarrow 1$ and $1 \rightarrow 1$, respectively. Since the number of subsets of any finite set A is $2^{|A|}$ and *different* pairs B/S define *different* rules, the number of different rules defined by two sets B and S is equal to $2^{k+1} \times 2^{k+1}$. Thus, the total number of k-valent rules described by the "birth"/"survival" lists is

$$N_{B/S,\,k} = 2^{2k+2}. \tag{22}$$

There is another way to characterize this type of local rules. Let us consider k-valent rules symmetric with respect to the group $\mathrm{Sym}\,(k)$ of all permutations of k outer points of the neighborhood. We shall call such rules k-symmetric. It is not difficult to count the total number of different q-ary k-symmetric rules:

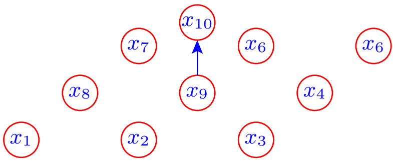
$$N_{q,\,\mathrm{Sym}(k)} = q^{\binom{k+q-1}{q-1}q}. \tag{23}$$

We see that (23) evaluated at $q = 2$ coincides with (22), i.e., $N_{2,\,\mathrm{Sym}(k)} = N_{B/S,\,k}$. Since k-valent Life rules are obviously k-symmetric we have the following

Proposition. *For any k the set of k-symmetric binary rules coincides with the set of k-valent Life rules.*

This proposition implies in particular that one can always express any k-symmetric binary rule in terms of the "birth"/"survival" lists.

The local relation of ***Conway's Life*** automaton R^{δ}_{CL} is defined on the 10-set $\delta = \{x_1, \ldots, x_{10}\}$:



Here the point $x_{10} \equiv x'_9$ is the next-time-step of the point x_9. By construction, elements of the 10-dimensional hypercube Σ^{δ} belong to the relation of ***Conway's Life*** automaton, i.e., $(x_1, \ldots, x_{10}) \in R^{\delta}_{CL}$, in the following cases:

1. $\left(\sum_{i=1}^{8} x_i = 3\right) \wedge (x_{10} = 1)$,
2. $\left(\sum_{i=1}^{8} x_i = 2\right) \wedge (x_9 = x_{10})$,
3. $x_{10} = 0$, if none of the above conditions holds.

The number of elements of the relation R^{δ}_{CL} is $\left|R^{\delta}_{CL}\right| = 512$. The relation R^{δ}_{CL}, as is the case for any cellular automaton, is *functional*: the state of x_{10} is uniquely determined by the states of other points. The state set $\Sigma = \{0, 1\}$ can be *additionally* endowed with the structure of the field $\mathbb{F}_2$. We accompany the below analysis of the structure of R^{δ}_{CL} by description in terms of polynomials from $\mathbb{F}_2\left[x_1, \ldots, x_{10}\right]$. This is done only for illustrative purposes and for comparison with the Gröbner basis method. In fact, we transform the relations into polynomials only for output. Transformation of any relation into polynomial form can be performed by computationally very cheap multivariate version of the Lagrange interpolation. In the case $q = 2$, the polynomial which set of zeros corresponds to a relation is constructed uniquely. If $q = p^n > 2$, there is a freedom in the choice of nonzero values of constructed polynomial, and the same relation can be represented by many polynomials.

The polynomial representing R^{δ}_{CL} takes the form

$$P_{CL} = x_{10} + x_9\left(\Pi_7 + \Pi_6 + \Pi_3 + \Pi_2\right) + \Pi_7 + \Pi_3, \tag{24}$$

where $\Pi_k \equiv \Pi_k\left(x_1, \ldots, x_8\right)$ is the kth *elementary symmetric polynomial* defined for n variables $x_1, \ldots, x_n$ by the formula:

$$\Pi_k\left(x_1, \ldots, x_n\right) = \sum_{1 \leq i_1 < i_2 < \cdots < i_k \leq n} x_{i_1} x_{i_2} \cdots x_{i_k}.$$

Hereafter, we will use the following notation:

$$\Pi_k \equiv \Pi_k\left(x_1, \ldots, x_8\right), \quad \Pi_k^i \equiv \Pi_k\left(x_1, \ldots, \widehat{x_i}, \ldots, x_8\right),$$
$$\Pi_k^{ij} \equiv \Pi_k\left(x_1, \ldots, \widehat{x_i}, \ldots, \widehat{x_j}, \ldots, x_8\right).$$

Applying the computer program to R_{CL}^{δ}, we find that the relation R_{CL}^{δ} is *reducible* and has the decomposition

$$R_{CL}^{\delta} = R_2^{\delta \backslash \{x_9\}} \bigcap \left(\bigcap_{k=1}^{7} R_1^{\delta \backslash \{x_{i_k}\}} \right), \tag{25}$$

where $(i_1, \ldots, i_7)$ is arbitrary 7-element subset of the set $(1, \ldots, 8)$. For brevity, we dropped in (25) the trivial factors $\Sigma^{\{x_{i_k}\}}$ entering into the general formula (20).

The eight relations $R_1^{\delta \backslash \{x_i\}}$ ($1 \leq i \leq 8$; for decomposition (25), it suffices to take any seven of them) have the following polynomial form:

$$x_9 x_{10}\left(\Pi_6^i + \Pi_5^i + \Pi_2^i + \Pi_1^i\right) + x_{10}\left(\Pi_6^i + \Pi_2^i + 1\right) + x_9\left(\Pi_7^i + \Pi_6^i + \Pi_3^i + \Pi_2^i\right) = 0.$$

Accordingly, the relation $R_2^{\delta \backslash \{x_9\}}$ has the form

$$x_{10}\left(\Pi_7 + \Pi_6 + \Pi_3 + \Pi_2 + 1\right) + \Pi_7 + \Pi_3 = 0.$$

The relations $R_1^{\delta \backslash \{x_i\}}$ and $R_2^{\delta \backslash \{x_9\}}$ are *irreducible* but *not prime*, and can be expanded in accordance with formula (19). Continuing the decomposition iterations, we finally obtain the following system of relations (in the polynomial form) that are satisfied for ***Conway's Life***:

$$\begin{aligned}
x_9 x_{10}\left(\Pi_2^i + \Pi_1^i\right) + x_{10}\left(\Pi_2^i + 1\right) + x_9\left(\Pi_7^i + \Pi_6^i + \Pi_3^i + \Pi_2^i\right) &= 0, && (26)\\
x_{10}\left(\Pi_3 + \Pi_2 + 1\right) + \Pi_7 + \Pi_3 &= 0, && (27)\\
\left(x_9 x_{10} + x_{10}\right)\left(\Pi_3^{ij} + \Pi_2^{ij} + \Pi_1^{ij} + 1\right) &= 0, && (28)\\
x_{10}\left(\Pi_3^i + \Pi_2^i + \Pi_1^i + 1\right) &= 0, && (29)\\
x_{10} x_{i_1} x_{i_2} x_{i_3} x_{i_4} &= 0. && (30)
\end{aligned}$$

One can easily interpret the simplest relations (30): if the point x_{10} is in the state 1, then at least one point in any set of four points surrounding x_9 must be in the state 0.

The above analysis of the relation R_{CL}^{δ} takes < 1 sec on a 1.8GHz AMD Athlon notebook with 960Mb.

To compute the Gröbner basis we must add to polynomial (24) ten polynomials

$$x_i^2 + x_i, \; i = 1, \ldots, 10$$

corresponding to the relation $x^q = x$ that holds for all elements of any finite field $\mathbb{F}_q$.

Computation of the Gröbner basis over $\mathbb{F}_2$ with the help of **Maple 9** gives the following:

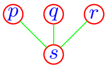
- pure lexicographic order with variable ordering $x_{10} \succ x_9 \succ \cdots \succ x_1$ does not provide any new information leaving initial polynomial (24) unchanged;

- pure lexicographic order with variable ordering $x_1 \succ x_2 \succ \cdots \succ x_{10}$ reproduces relations (26)—(30) (modulo several polynomial reductions violating the symmetry of polynomials); the computation takes 1 h 22 min;

- degree-reverse-lexicographic order also reproduces system (26)—(30) (same comment as above); the times are: 51 min for the variable ordering $x_1 \succ x_2 \succ \cdots \succ x_{10}$, and 33 min for the ordering $x_{10} \succ x_9 \succ \ldots \succ x_1$.

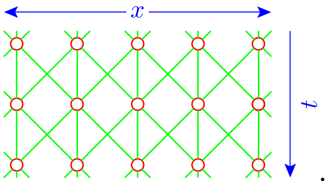
3.2.2. Elementary Cellular Automata

Simplest binary, nearest-neighbor, 1-dimensional cellular automata were named *elementary cellular automata* by S. Wolfram, who has extensively studied their properties [16]. A large collection of results concerning these automata is presented in Wolfram's online atlas [17]. In the exposition below we use Wolfram's notations and terminology. The elementary cellular automata are simpler than ***Conway's Life***, and we may pay more attention to the topological aspects of our approach.

Local rules of the elementary cellular automata are defined on the 4-set $\delta = \{p, q, r, s\}$ which can be pictured by the icon  . A local rule is a binary function of the form $s = f(p, q, r)$. There are totally $2^{2^3} = 256$ such functions, each of which can be indexed with an 8-bit binary number.

Our computation with relations representing the local rules shows that the total number 256 of them is divided into 118 reducible and 138 irreducible relations. Only two of the irreducible relations appeared to be prime, namely, the rules 105 and 150 in Wolfram's numeration. This numeration is based on the "big-endian" — i.e., opposite to our convention (21) — representation of binary numbers. Note, that the prime rules 105 and 150 have linear polynomial forms: $s = p + q + r + 1$ and $s = p + q + r$, respectively.

We consider the elementary automata on a space-time lattice with integer coordinates (x, t), i.e., $x \in \mathbb{Z}$ or $x \in \mathbb{Z}_m$ (spatial m-periodicity), $t \in \mathbb{Z}$. We denote a state of the point on the lattice by $u(x, t) \in \Sigma = \{0, 1\}$. Generally the points are connected as is shown in the picture

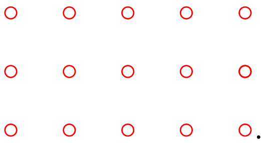


.

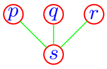
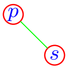
The absence of horizontal ties expresses the independence of "space-like" points in cellular automata.

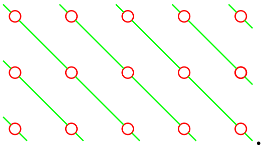
Reducible Automata. The analysis shows that some automata with reducible local relations can be represented as unions of automata defined on disconnected subcomplexes:

- Two automata 0 and 255 are determined by unary relations $s = 0$ and $s = 1$ on the disconnected set of points:



Note that unary relations are usually called *properties*.

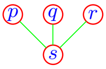
- Six automata 15, 51, 85, 170, 204 and 240 are, in fact, disjoint collections of spacially zero-dimensional automata, i.e., single cells evolving in time. As an example, let us consider the automaton 15. The local relation is defined on the set  and its bit table is 0101010110101010. This relation is reduced to the relation on the face  with bit table 0110. The spacetime lattice is split in the following way:

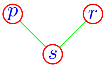


The bit table 0110 means that the points p and s can be only in opposite states, and we can write immediately the general solution for the automaton 15:

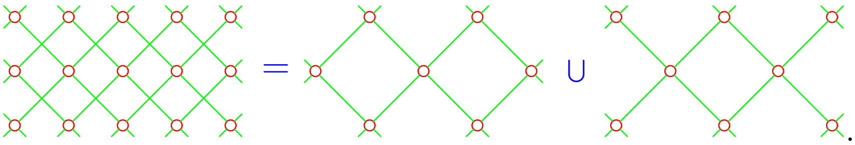
$$u(x,t) = a(x-t) + t \mod 2,$$

where $u(x,0) \equiv a(x)$ is an arbitrary initial condition.

- Each of the ten automata 5, 10, 80, 90, 95, 160, 165, 175, 245, 250 is decomposed into two identical automata. As an example let us consider the rule 90. This automaton is distinguished as producing the fractal of topological dimension 1 and Hausdorff dimension $\ln 3/\ln 2 \approx 1.58$ known as the *Sierpinski sieve* (or *gasket* or *triangle*). Its local relation on the set  is represented by the bit table 1010010101011010. The relation is reduced to the relation with the bit table

10010110 on the face . (31)

It can be seen from the structure of face (31) that the spacetime lattice is split into two identical independent complexes as is shown



To find a general solution of the automaton 90 it is convenient to use the polynomial form of relation (31) $s+p+r=0$. With this linear expression, the general solution is easily constructed:

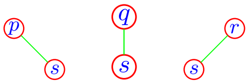
$$u(x,t)=\sum_{k=0}^{t}\binom{t}{k}a(x-t+2k)\mod 2,\qquad u(x,0)\equiv a(x).$$

Using Proper Consequences. Proper consequences — even if they are not functional — can provide useful information on the behavior of a cellular automaton.

For example, 64 automata[10] (with both reducible and irreducible local relations) have proper consequences with the bit table

$$1101 \tag{32}$$

on, at least, one of the faces



. (33)

The algebraic forms of relation (32) on faces (33) are $ps+s=0,\ \ qs+s=0,\ \ rs+s=0,$ respectively.

Relation (32) is *not functional*, and hence can not describe any deterministic evolution. Nevertheless, it imposes severe restrictions on the behavior of the automata having such proper consequences. The features of the behavior resulting from relation (32) are clearly seen from many of computational results presented in the atlas [17]. A typical pattern from this atlas is reproduced in Fig. 3, where several evolutions of the automaton 168 are presented. In the figure, 0's and 1's are denoted by the empty and filled square cells,

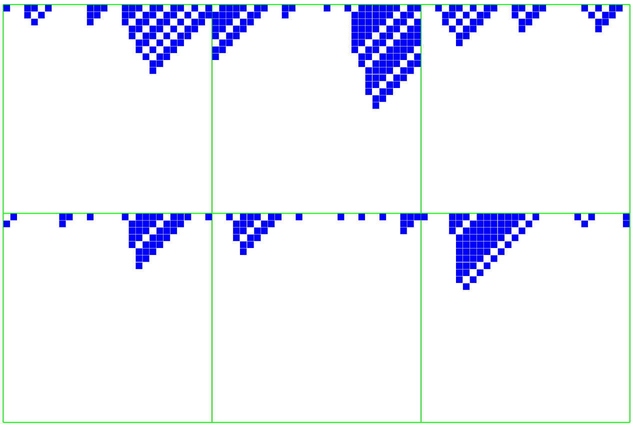


Figure 3. Rule 168. Several random initial conditions

[10] The complete list of these automata in Wolfram's numeration is as follows: 2, 4, 8, 10, 16, 32, 34, 40, 42, 48, 64, 72, 76, 80, 96, 112, 128, 130, 132, 136, 138, 140, 144, 160, 162, 168, 171, 174–176, 186, 187, 190–192, 196, 200, 205, 206, 208, 220, 222–224, 234–239, 241–254.

respectively. Note that the authors of the figure use a spatially periodic condition: $x \in \mathbb{Z}_{30}$.

The local relation of the automaton 168 — its polynomial form is $pqr + qr + pr + s = 0$ — has the proper consequence $rs + s = 0$. Relation (32) means that if, say r, as for the rule 168, is in the state 1 then s may be in both states 0 or 1, but if the state of r is 0, then the state of s must be 0:

$$r = 1 \Rightarrow s = 0 \vee s = 1,$$

$$r = 0 \Rightarrow s = 0.$$

One can see that all evolutions in Fig. 3 consist of diagonals $x = x_0 - t$ directed leftward and downward. Each diagonal begins with a several units, but after the first appearance of zero all subsequent points along the diagonal are zeros.

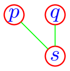
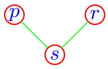
Canonical Decomposition vs. Gröbner Basis. In this paragraph we compare our canonical decomposition (19) with Gröbner basis in the polynomial case. Let us begin with two examples of elementary cellular automata. The Gröbner bases are computed in the total degree and reverse lexicographical order of monomials. The trivial polynomials $p^2 + p$, $q^2 + q$, $r^2 + r$ and $s^2 + s$ are omitted in the Gröbner bases descriptions.

- **Automaton 30** is remarkable by its chaotic behavior and is even used as a random number generator in ***Mathematica***.

  Relation: 1001010101101010 or $qr + s + r + q + p = 0$.

  Canonical Decomposition:

  Proper consequences:

face		
bit table	11011110	11011110
polynomial	$qs + pq + q$	$rs + pr + r.$

  Principal factor: 1011111101111111 or $qrs + pqr + rs + qs + pr + pq + s + p = 0$.

  Gröbner basis: $\{qr + s + r + q + p,\ qs + pq + q,\ rs + pr + r\}$.

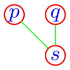
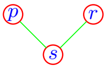
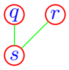
  Thus for the rule 30 the polynomials of the canonical decomposition coincide (modulo obvious polynomial substitutions) with the Gröbner basis.

- **Automaton 110** is, like a Turing machine, *universal*, i.e., it can simulate any computational process, in particular, any other cellular automaton.

  Relation: 1100000100111110 or $pqr + qr + s + r + q = 0$.

  Canonical Decomposition:

  Proper consequences:

face			
bit table	11011111	11011111	10010111
polynomial	$pqs + qs + pq + q$	$prs + rs + pr + r$	$qrs + s + r + q.$

Principal factor: 1111111111111110 or $pqrs = 0$.

Gröbner basis:

$$\{prs + rs + pr + r,\ qs + rs + r + q,\ qr + rs + s + q,\ pr + pq + ps\}\,.$$

For automaton 110, the polynomials of the Gröbner basis are not identical with the polynomials of the canonical decomposition. The system of relations defined by the Gröbner basis is:

$$\begin{aligned}
R_1^{\{p,r,s\}} &= 11011111 = (prs + rs + pr + r = 0)\,,\\
R_2^{\{q,r,s\}} &= 10011111 = (qs + rs + r + q = 0)\,,\\
R_3^{\{q,r,s\}} &= 10110111 = (qr + rs + s + q = 0)\,,\\
R_4^{\{p,q,r,s\}} &= 1110101110111110 = (pr + pq + ps = 0)\,.
\end{aligned}$$

In general, the folowing differences between our approach and the Gröbner basis method can be mentioned.

- In contrast to a Gröbner basis, a base relation, defined as intersection of conditions, agrees with the standard in logic and set theory notion of compatibility.
- In contrast to a canonical decomposition a Gröbner basis may look beyond the polynomial context as a collection of accidental supersets.
- There is some analogy between Gröbner bases and canonical decompositions — in fact, they coincide in about half of cases in our computations.
- Canonical decomposition is more efficient for problems with polynomials of arbitrary degree — the above computation with Conway's automaton is an example.
- For small degree problems with large number n of indeterminates the Gröbner basis outperforms canonical decomposition — the number of polynomials of bounded degree is a polynomial function of n, whereas the algorithm of canonical decomposition scans exponential number q^n of the hypercube points.

4. Soliton-like Structures in Deterministic Dynamics

Symmetries of deterministic systems impose severe restrictions on the system dynamics [6]. In particular, for the first order[11] functional relations:

[11] This means that evolution relation (1) takes the form $s_t = F\left(s_{t-1}\right)$.

- *dynamical trajectories* pass group orbits in *non-decreasing* order of orbit sizes,
- *periodic* trajectories lie within orbits of the *same size*.

One of the characteristic features of dynamical systems with non-trivial symmetries is formation of moving form-preserving structures.

Let us begin with a simple example. Consider a cube X whose vertices take values in two-element set, say $\Sigma = \{0, 1\}$. By the way, as is clear from Fig. 4, a cube can be interpreted as a simplest “finite model of graphene”. The 48-element symmetry group of a cube has the structure $\mathrm{G} = \mathbb{Z}_2 \times \mathrm{Sym}\left(4\right)$. The group is generated by 3 elements:

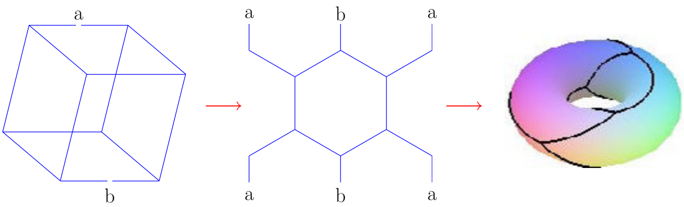


Figure 4. The graph of cube forms 4-gonal (6 tetragons) lattice in sphere $\mathbb{S}^2$ and 6-gonal (4 hexagons) lattice in torus $\mathbb{T}^2$.

1. 120^o rotation around diagonal of the cube;
2. 90^o rotation around axis passing through the centers of opposite cube faces;
3. reflection interchanging opposite faces of the cube.

Total number of states of the model is $\left|\Sigma^{\mathrm{X}}\right| = 2^8 = 256$. If we assume that the group Γ is trivial, then $\mathrm{W} = \Gamma^{\mathrm{X}} \rtimes \mathrm{G} = 1 \rtimes \mathrm{G} \cong \mathrm{G}$. The group W splits the set Σ^{X} into 22 orbits in accordance with the table:

Size of orbits	1	2	4	6	8	12	24
Number of orbits	2	1	2	2	5	4	6

.

Let us consider a *deterministic* dynamical system on the cube, namely, symmetric binary 3-valent cellular automaton with the rule 86. The number 86 is the “little endian” representation of the bit string 01101010 taken from the last column of the rule table with $\mathrm{Sym}\left(3\right)$-symmetric combinations of values for $x_{1,i}, x_{2,i}, x_{3,i}$

$x_{1,i}$	$x_{2,i}$	$x_{3,i}$	x_i	x_i'
0	0	0	0	0
0	0	0	1	1
1	0	0	0	1
1	0	0	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

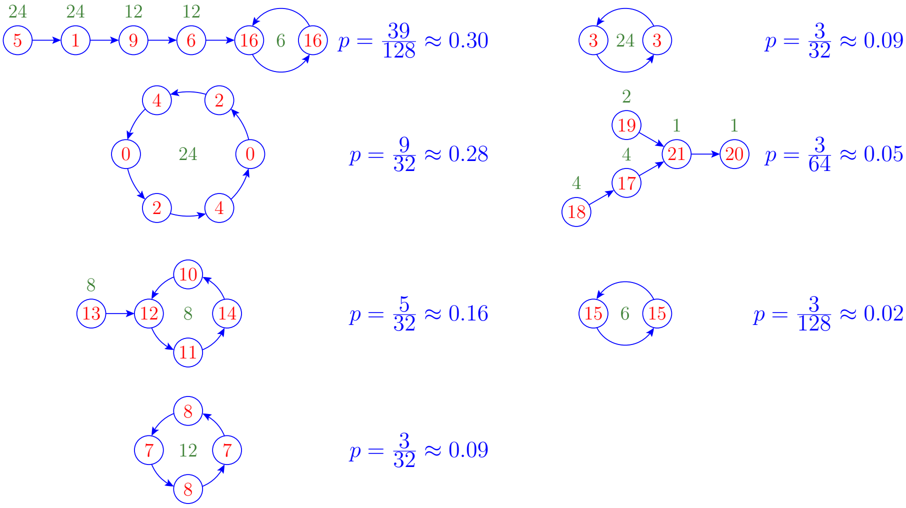
.

Here x_i is value of ith vertex of the cube; $x_{1,i}, x_{2,i}, x_{3,i}$ are values of the cube vertices adjacent to the ith one and x'_i is the next time value of ith vertex. The rule can also be represented in ***Conway's Life*** style "Birth"/"Survival" notation as B123/S0, or as polynomial over the field $\mathbb{F}_2$

$$x'_i = x_i + \Pi_3 + \Pi_2 + \Pi_1,$$

where $\Pi_1 = x_{1,i} + x_{2,i} + x_{3,i}$, $\Pi_2 = x_{1,i}x_{2,i} + x_{1,i}x_{3,i} + x_{2,i}x_{3,i}$, $\Pi_3 = x_{1,i}x_{2,i}x_{3,i}$ are elementary symmetric functions.

The phase portrait of the automaton is shown in Fig. 5, where the group orbits are represented by circles containing the ordinal numbers[12] of orbits within. The numbers over orbits and within cycles are sizes of the orbits (recall that all orbits belonging to the same cycle have equal sizes — see the beginning of this section). The rational number p indicates the *weight* of the corresponding element of the phase portrait. In fact, p is a probability for randomly chosen state to appear in an isolated cycle or to be caught by an attractor: p = (*size of basin*)/(*total number of states*). Here *size of basin* is sum of sizes of orbits involved in the struture.



24
24
12
12
5
1
9
6
16
6
16
$p = \frac{39}{128} \approx 0.30$
3
24
3
$p = \frac{3}{32} \approx 0.09$
4
2
0
24
0
2
4
$p = \frac{9}{32} \approx 0.28$
2
19
1
1
4
21
20
4
17
18
$p = \frac{3}{64} \approx 0.05$
8
13
10
12
8
14
11
$p = \frac{5}{32} \approx 0.16$
15
6
15
$p = \frac{3}{128} \approx 0.02$
8
7
12
7
8
$p = \frac{3}{32} \approx 0.09$

Figure 5. Rule 86. Equivalence classes of trajectories on hexahedron.

Generalizing this example, we see that if the symmetry group W splits the state set Σ^{X} of *deterministic* dynamical system into *finite* number of orbits, then after some lapse of time *any* trajectory comes *inevitably* to a cycle over some finite sequence of orbits. This just means formation of *soliton-like structures*. Namely, let us consider evolution

$$\sigma_{t_0}(x) \rightarrow \sigma_{t_1}(x) = A_{t_1 t_0}\left(\sigma_{t_0}(x)\right). \tag{34}$$

If the states at the moments t_0 and t_1 belong to the *same orbit*: $\sigma_{t_0}(x) \in O_i$ and $\sigma_{t_0}(x) \in O_i$, $O_i \subseteq \Sigma^{\mathrm{X}}$; then evolution (34) can be replaced by the *group action*

$$\sigma_{t_1}(x) = \sigma_{t_0}(x)w, \quad w \in W,$$

[12] These numbers are specified by the computer program in the course of computation.

i.e., the initial state ("shape") $\sigma_{t_0}(x)$ is reproduced after some "*movement*" in the space Σ^{X}.

The following are several examples (including continuous cases) of cycles over group orbits:

- *traveling waves* $\sigma(x - vt)$ in mathematical physics — the Galilei group;
- "*generalized coherent states*" in quantum physics — unitary representations of compact Lie groups;
- "*spaceships*" in cellular automata — lattice symmetries.

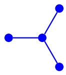
Let us consider the "glider" — one of the "spaceships" in ***Conway's Life*** automaton.

The space X of ***Conway's Life*** is a square lattice. For the finiteness, we shall assume that the lattice is closed into the $N \times N$ torus. In the general case $N \neq 4$ the symmetry group of X is the semidirect product of two-dimensional translations $\mathrm{T}^2 = \mathbb{Z}_N \times \mathbb{Z}_N$ and the dihedral group $\mathrm{D}_8 = \mathbb{Z}_4 \rtimes \mathbb{Z}_2$:

$$\mathrm{G} = \mathrm{T}^2 \rtimes \mathrm{D}_8, \quad \text{if} \quad N = 3, 5, 6, \ldots, \infty. \qquad (35)$$

In the case $N = 4$ the translation subgroup $\mathrm{T}^2 = \mathbb{Z}_4 \times \mathbb{Z}_4$ is *not normal* and G has a bit more complicated structure [7]:

$$\mathrm{G} = \overbrace{((((\mathbb{Z}_2 \times \mathrm{D}_8) \rtimes \mathbb{Z}_2) \rtimes \mathbb{Z}_3) \rtimes \mathbb{Z}_2)}^{\text{normal closure of } \mathrm{T}^2} \rtimes \mathbb{Z}_2. \qquad (36)$$

The extra symmetry $\mathbb{Z}_3$ in (36) can be explained by the $\mathbb{Z}_3$ symmetry of the four-vertex Dynkin diagram $D_4 =$  associated with the case $N = 4$.

The set of local (cell) states of ***Conway's Life*** is $\Sigma = \{\text{"dead"}, \text{"alive"}\} = \{0, 1\}$. Since the local rule of ***Conway's Life*** is not symmetric with respect to the transposition $0 \leftrightarrow 1$ of the local states, the internal symmetry group is trivial, i.e., $\Gamma = \{\mathbf{1}\}$ and hence $\Gamma^{\mathrm{X}} = \{\mathbf{1}\}$. Thus, we have $\mathrm{W} = \Gamma^{\mathrm{X}} \rtimes \mathrm{G} = \mathbf{1} \rtimes \mathrm{G} \cong \mathrm{G}$. The natural action of W on functions $\sigma(x) \in \Sigma^{\mathrm{X}}$ takes the form $\sigma(x)w = \sigma\left(xg^{-1}\right)$, where $w = (\mathbf{1}, g)$, $g \in \mathrm{G}$.

Fig. 6 shows four steps of evolution of the glider. The figure demonstrates how the evolution is reduced to the group action. $N > 4$ is assumed.

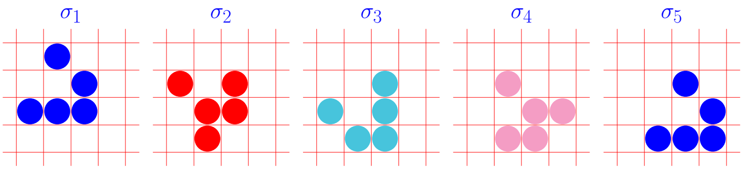


Figure 6. Example of soliton-like structure. "Glider" in ***Conway's Life*** is cycle in *two* orbits of the group $\mathrm{G} = \mathrm{T}^2 \rtimes \mathrm{D}_8$: configurations σ_3 and σ_4 are obtained from σ_1 and σ_2, respectively, by *the same* combination of downward *shift*, 90^o clockwise *rotation* and *reflection* in respect to vertical.

Comments on Reversibility in Discrete Systems.

A typical deterministic dynamical system is *irreversible* — it's phase portrait *modulo* group orbits looks like in Fig. 5. We see there several isolated and limit cycles (fixed points are regarded as cycles of unit length) accompanied by influxes flowing into the limit cycles. In contrast to continuous systems, any discrete system "forgets" influxes after some time and appears in either isolated or limit cycles. After loss of information about influxes both types of cycles became physically indistinguishable and the system behaves just like reversible. This might be a hint for explanation of observable reversibility of the fundamental laws of nature.

In this connection we would like to mention recent works of G. 't Hooft. One of the difficulties of quantum gravity is a conflict between irreversibility of gravity — information loss at the black hole horizon — with reversibility and unitarity of the standard quantum mechanics. In several papers of recent years (see, e.g., [18, 19]) 't Hooft developed an approach to reconciling both theories. The approach is based on the following assumptions

- physical systems have *discrete degrees of freedom* at tiny (Planck) distance scales;
- the states of these degrees of freedom form *primordial* basis of Hilbert space (with nonunitary evolution);
- primordial states form *equivalence classes*: two states are equivalent if they evolve into the same state after some lapse of time;
- the equivalence classes by construction form basis of Hilbert space with unitary evolution described by time-reversible Schrödinger equation.

In our terminology this corresponds to transition to limit cycles: in a finite time of evolution the limit cycle becomes physically indistinguishable from reversible isolated cycle — the system "forgets" its pre-cycle history.

This type of irreversibility hardly can be observed experimentally (assuming, of course, that considered models may have at all any relation to physical reality). The system should probably spend time of order the Planck unit ($\approx 10^{-44}$ sec) out of a cycle and potentially infinite time on the cycle. Nowadays, the shortest experimentally fixed time is about 10^{-18} sec or 10^{26} Planck units.

5. Mesoscopic Lattice Models

Discrete symmetry analysis simplifies manipulations with *microcanonical ensembles* and search of *phase transitions*. This allows to reveal subtle details in behavior of *mesoscopic models*.

5.1. Statistical Mechanics

As we mentioned earlier, the state of deterministic dynamical system at any point of time is determined uniquely by previous states of the system. A Markov chain — for which transition from any state to any other is possible with some probability — is a typical example of *non-deterministic* dynamical system. In this section we apply symmetry approach to the lattice models in statistical mechanics. These models can be regarded as special instances of Markov chains. *Stationary distributions* of the Markov chains are studied by the methods of statistical mechanics.

The main tool of conventional statistical mechanics is the Gibbs *canonical ensemble* — imaginary collection of identical systems placed in a huge thermostat with temperature T. The statistical properties of canonical ensemble are encoded in the *canonical partition function*

$$Z = \sum_{\sigma\in\Sigma^{\mathrm{X}}} \mathrm{e}^{-E_\sigma/k_B T}. \tag{37}$$

Here Σ^{X} is the set of microstates, E_σ is energy of microstate σ, k_B is Boltzmann's constant. The canonical ensemble is essentially asymptotic concept: its formulation is based on approximation called “thermodynamic limit”. For this reason, the canonical ensemble approach is applicable only to large (strictly speaking, infinite) homogeneous systems.

5.2. Mesoscopy

Nowadays much attention is paid to study systems which are too large for a detailed microscopic description but too small for essential features of their behavior to be expressed in terms of classical thermodynamics. This discipline —often called *mesoscopy* — covers wide range of applications from nuclei, atomic clusters and nanotechnological structures to multi-star systems [20, 21, 22]. To study *mesoscopic* systems one should use more fundamental *microcanonical ensemble* instead of canonical one. A microcanonical ensemble is a collection of identical isolated systems at fixed energy. Its definition does not include any approximating assumptions. In fact, the only key assumption of a microcanonical ensemble is that all its microstates are equally probable. This leads to the *entropy* formula

$$S_E = k_B \ln \Omega_E, \tag{38}$$

or, equivalently, to the *microcanonical partition function*

$$\Omega_E = \mathrm{e}^{S_E/k_B}. \tag{39}$$

Here Ω_E is the number of microstates at fixed energy $E : \sum\limits_{E}\Omega_E = \left|\Sigma^{\mathrm{X}}\right|$. In what follows we will omit Boltzmann's constant assuming $k_B = 1$. Note that in the thermodynamic limit the microcanonical and canonical descriptions are equivalent and the link between them is provided by the Laplace transform. On the other hand, mesoscopic systems demonstrate experimentally and computationally observable peculiarities of behavior like heat flows from cold to hot, negative specific heat or “convex intruders” in the entropy versus energy diagram, etc. These anomalous — from the point of view of canonical thermostatistics — features have natural explanation within microcanonical statistical mechanics [22].

5.2.1. Lattice Models.

In this section we apply symmetry analysis to study mesoscopic lattice models. Our approach is based on exact enumeration of group orbits of microstates. Since statistical studies are based essentially on different simplifying assumptions, it is important to control these assumptions by exact computation, wherever possible. Moreover, we might hope to reveal subtle details in behavior of system under consideration with the help of exact computation.

As an example, let us consider the Ising model. The model consists of *spins* placed on a lattice. The set of vertex values is $\Sigma = \{-1, 1\}$ and the interaction Hamiltonian is given by

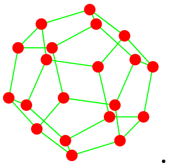
$$H = -J\sum_{(i,j)} s_i s_j - B\sum_i s_i, \tag{40}$$

where $s_i, s_j \in \Sigma$; J is a coupling constant ($J > 0$ and $J < 0$ correspond to *ferromagnetic* and *antiferromagnetic* cases, respectively); the first sum runs over all edges (i, j) of the lattice; B is an external "magnetic" field. The second sum $M = \sum\limits_i s_i$ is called the *magnetization*. To avoid unnecessary technical details we will consider only the case $J > 0$ (assuming $J = 1$) and $B = 0$ in what follows.

Let us remind that if the local symmetry group $\mathrm{G_{loc}}$ decomposes the sets of edges of lattice neighborhoods into nontrivial orbits, then the interaction Hamiltonian should be modified (see, e.g., Eq. (8) on page 127).

Since Hamiltonian and magnetization are constants on the group orbits, we can count numbers of microstates corresponding to particular values of these functions – and hence compute all needed statistical characteristics – simply by summation of sizes of appropriate orbits.

Fig. 7 shows microcanonical partition function for the Ising model on the dodecahedron



Here total number of microstates $\left|\Sigma^{\mathrm{X}}\right| = 1048576$, number of lattice vertices $N_{\mathrm{X}} = 20$, energy E is value of Hamiltonian.

Of course, other characteristics of the system can be computed easily in this way.

5.3. Phase Transitions

Needs of nanotechnological science and nuclear physics attract special attention to phase transitions in finite systems. Unfortunately classical thermodynamics and the rigorous theory of critical phenomena in homogeneous infinite systems fails at the mesoscopic level. Several approaches have been proposed to identify phase transitions in mesoscopic systems. Most accepted of them is search of *"convex intruders"* [23] in the entropy versus energy diagram. In the standard thermodynamics there is a relation

$$\left.\frac{\partial^2 S}{\partial E^2}\right|_V = -\frac{1}{T^2}\frac{1}{C_V}, \tag{41}$$

where C_V is the specific heat at constant volume.

Relation (41) implies that $\partial^2 S/\partial E^2\big|_V < 0$ and hence the entropy versus energy diagram must be concave. Nevertheless, in mesoscopic systems there might be intervals of

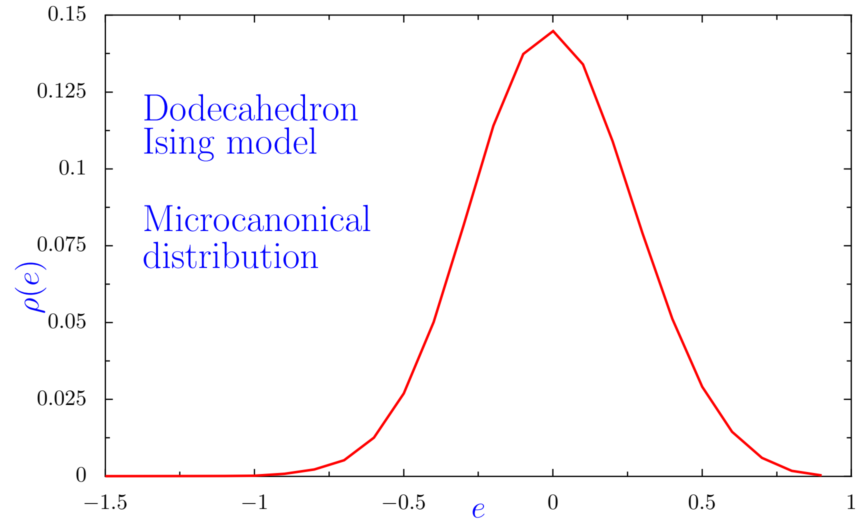


Figure 7. Ising model on dodecahedron. Microcanonical distribution.

energy where $\partial^2 S/\partial E^2\big|_V > 0$. These intervals correspond to first-order phase transitions and are called *"convex intruders"*. From the point of view of standard thermodynamics one can say about phenomenon of *negative heat capacity*, of course, if one accepts that it makes sense to define the variables T and C_V as temperature and the specific heat at these circumstances. In [24] it was demonstrated via computation with exactly solvable lattice models that the convex intruders flatten and disappear in the models with local interactions as the lattice size grows, while in the case of long-range interaction these peculiarities survive even in the limit of an infinite system (both finite and long-range interacting infinite systems are typical cases of systems called *nonextensive* in statistical mechanics).

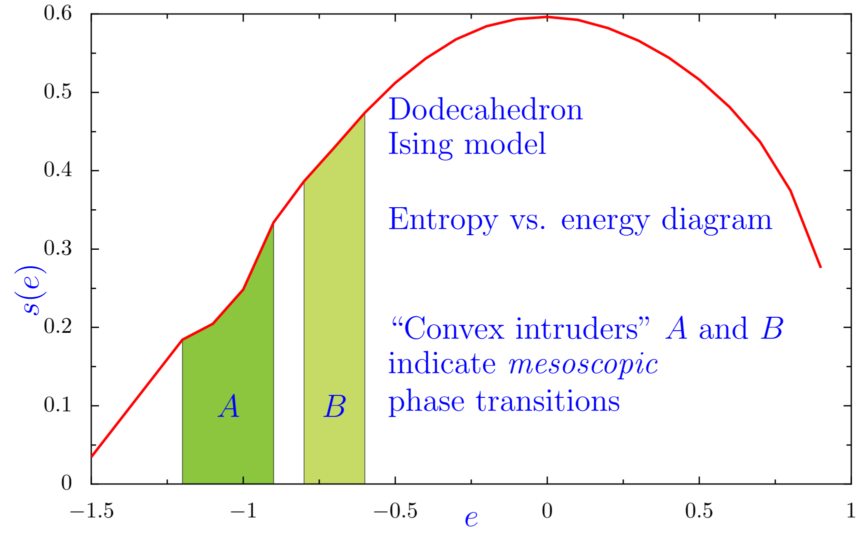


Figure 8. Ising model on dodecahedron. "Convex intruders" on entropy vs. energy diagram indicate *mesoscopic* phase transitions.

A convex intruder can be found easily by computer for the discrete systems we discuss here. Let us consider three adjacent values of energy E_{i-1}, E_i, E_{i+1} and corresponding numbers of microstates $\Omega_{E_{i-1}}, \Omega_{E_i}, \Omega_{E_{i+1}}$. In our discrete case the ratio $\frac{E_{i+1}-E_i}{E_i-E_{i-1}}$ is always rational number p/q and we can write the convexity condition for entropy in terms of numbers of microstates as easily evaluated inequality

$$\Omega_{E_i}^{p+q} < \Omega_{E_{i-1}}^{p} \Omega_{E_{i+1}}^{q}. \tag{42}$$

As a rule $E_{i+1} - E_i = E_i - E_{i-1}$ and inequality (42) takes the form

$$\Omega_{E_i}^{2} < \Omega_{E_{i-1}} \Omega_{E_{i+1}}.$$

This form means that within convex intruder the number of states with the energy E_i is less than *geometric mean* of numbers of states at the neighboring energy levels.

Fig. 8 shows the entropy vs. energy diagram for the Ising model on dodecahedron. The diagram has apparent convex intruder A in the specific energy interval $[-1.2, -0.9]$. Exact computation reveals also a subtle convex intruder B in the interval $[-0.8, -0.6]$.

6. Gauge Connection and Quantization

All most successful contemporary theories in fundamental physics are gauge theories. There are also numerous applications of gauge theories in mathematics (topological quantum field theory, invariants of 3- and 4-manifolds, monoidal categories, Hopf algebras and quantum groups, etc. [25]).

In fact, the gauge principle expresses the very general idea that in spite of the fact that any observable data are represented in different "reference frames" at different points in space[13] and time, there should be some way to compare these data.

6.1. Discrete Gauge Principle

At the set-theoretic level, i.e., in the form suitable for both discrete and continuous cases, the main concepts of the gauge principle can be reduced to the following. We have

- a set $\mathcal{T}$, discrete or continuous *time*, $\mathcal{T} \cong \mathbb{Z}$ or $\mathcal{T} \cong \mathbb{R}$;
- a set X, *space*;
- the sets $\mathcal{T}$ and X are combined into a space-time $\mathcal{M} = \mathrm{X} \times \mathcal{T}$;
- a set Σ, *local states*;
- a group $\Gamma \leq \mathrm{Sym}(\Sigma)$ acting on Σ, *internal symmetries*;
- identification of data describing the states from Σ makes sense only *modulo* symmetries from Γ — this is arbitrariness in the choice of a "*reference frame*";

[13] Consideration only time evolution of general set of states $\mathcal{S}$ leads to the trivial gauge structures. Gauge theories of interest are possible if there exists underlying space structure, i.e., $\mathcal{S} = \Sigma^{\mathrm{X}}$.

- there is no *a priori* connection between data (i.e., between reference frames) at different points $x,\ y \in \mathcal{M}$ — we should impose this *connection* (or *parallel transport*) explicitly as Γ-valued function on edges (pairs of points) of abstract graph:

$$\varsigma(y) = \sigma(x)\pi\left(x, y\right), \ \ \pi\left(x, y\right) \in \Gamma, \ \ \sigma(x), \varsigma(y) \in \Sigma;$$

  the connection $\pi\left(x, y\right)$ has the obvious property $\pi\left(x, y\right) = \pi\left(y, x\right)^{-1}$;

- a connection $\widetilde{\pi}\left(x, y\right)$ is called *trivial* if it can be expressed in terms of a function on *vertices* of the graph: $\widetilde{\pi}\left(x, y\right) = p\left(x\right)p\left(y\right)^{-1}, \ \ p\left(x\right), p\left(y\right) \in \Gamma$;

- invariance with respect to the gauge symmetries depending on time and space leads to the transformation rule for connection

$$\pi(x, y) \rightarrow \gamma(x)^{-1}\pi(x, y)\gamma(y), \ \ \gamma(x), \gamma(y) \in \Gamma; \tag{43}$$

- the *curvature* of connection $\pi(x, y)$ is defined as the conjugacy class[14] of the *holonomy* along a cycle of a graph:

$$\pi(x_1, x_2, \ldots, x_k) = \pi(x_1, x_2)\pi(x_2, x_3)\cdots\pi(x_k, x_1);$$

  the curvature of trivial connection is obviously trivial: $\widetilde{\pi}(x_1, \ldots, x_k) = \mathbf{1}$;

- the gauge principle does not tell us anything about the evolution of the connection itself, so gauge invariant relation describing dynamics of connection (*gauge field*) should be added.

Let us give two illustrations of how these concepts work in continuous case.

Electrodynamics. Abelian prototype of all gauge theories

Here the set $\mathcal{M}$ is 4-dimensional Minkowski space with points $x = (x^\mu)$ and the set of states is Hilbert space of complex scalar (Schrödinger equation) or spinor (Dirac equation) fields $\psi(x)$. The symmetry group of the Lagrangians and physical observables is the unitary group $\Gamma = \mathrm{U}(1)$. The elements of Γ^{X} can be represented as $e^{-i\alpha(x)}$.

Let us consider the parallel transport for two closely situated space-time points:

$$\pi(x, x + \Delta x) = e^{-i\rho(x, x+\Delta x)}.$$

Specializing transformation rule (43) to this particular case

$$\pi'(x, x + \Delta x) = e^{i\alpha(x)}\pi(x, x + \Delta x)e^{-i\alpha(x+\Delta x)},$$

substituting approximations

$$\pi(x, x + \Delta x) = e^{-i\rho(x, x+\Delta x)} \approx \mathbf{1} - iA(x)\Delta x,$$

$$\pi'(x, x + \Delta x) = e^{-i\rho(x, x+\Delta x)} \approx \mathbf{1} - iA'(x)\Delta x,$$

[14] The conjugacy equivalence means that $\pi'(x_1, \ldots, x_k) \sim \gamma^{-1}\pi(x_1, \ldots, x_k)\gamma$ for any $\gamma \in \Gamma$.

$$e^{-i\alpha(x+\Delta x)} \approx e^{-i\alpha(x)}\left(\mathbf{1} - i\nabla\alpha(x)\Delta x\right),$$

and taking into account commutativity of $\Gamma = \mathrm{U}(1)$ we obtain

$$A'(x) = A(x) + \nabla\alpha(x) \quad \text{or, in components,} \quad A'_\mu(x) = A_\mu(x) + \frac{\partial\alpha(x)}{\partial x^\mu}. \tag{44}$$

The 1-form A taking values in the Lie algebra of $\mathrm{U}(1)$ and its differential $F = (F_{\mu\nu}) = \mathrm{d}A$ are identified with the electromagnetic *vector potential* and the *field strength*, respectively. To provide the gauge invariance of the equations for field $\psi(x)$ we should replace partial by covariant derivatives

$$\partial_\mu \to D_\mu = \partial_\mu - iA_\mu(x)$$

in those equations.

Finally, evolution equations for the gauge field $A(x)$ should be added. In the case of electromagnetics these are Maxwell's equations:

$$\mathrm{d}F = 0 \qquad \textit{first pair} \tag{45}$$

$$\mathrm{d}\star F = 0 \qquad \textit{second pair}. \tag{46}$$

Here $\star$ is the *Hodge conjugation* (*Hodge star operator*). Note that equation (46) corresponds to *vacuum Maxwell's equations*. In the presence of the *current* J the *second pair* takes the form $\star\,\mathrm{d}\star F = J$. Note also that the *first pair* is essentially *a priori* statement, it reflects simply the fact that F, by definition, is the differential of an exterior form.

Non-Abelian Gauge Theories in Continuous Space-time

Only minor modifications are needed for the case of non-Abelian Lie group Γ. Again expansion of the Γ-valued parallel transport for two close space-time points x and $x + \Delta x$ with taking into account that $\pi(x,x) = 1$ leads to introduction of a Lie algebra valued 1-form $A = (A_\mu)$:

$$\pi(x, x+\Delta x) \approx \mathbf{1} + A_\mu(x)\Delta x^\mu.$$

Infinitesimal manipulations with formula (43)

$$\gamma(x)^{-1}\pi(x,x+\Delta x)\gamma(x+\Delta x) \;\longrightarrow\; \gamma(x)^{-1}\left(\mathbf{1} + A_\mu(x)\Delta x^\mu\right)\left(\gamma(x) + \frac{\partial\gamma(x)}{\partial x^\mu}\Delta x^\mu\right)$$

lead to the following transformation rule

$$A'_\mu(x) = \gamma(x)^{-1}A_\mu(x)\gamma(x) + \gamma(x)^{-1}\frac{\partial\gamma(x)}{\partial x^\mu}. \tag{47}$$

The curvature 2-form

$$F = dA + [A \wedge A]$$

is interpreted as the *physical strength field*. In particular, the *trivial* connection

$$\widetilde{A}_\mu(x) = \gamma_0(x)^{-1}\frac{\partial\gamma_0(x)}{\partial x^\mu}$$

is *flat*, i.e., its curvature $F = 0$.

There are different approaches to construct dynamical equations for gauge fields [25]. The most important example is *Yang-Mills theory* based on the Lagrangian

$$L_{YM} = \mathrm{Tr}\left[F \wedge \star F\right].$$

The Yang-Mills equations of motion read

$$\begin{aligned} \mathrm{d}F + [A \wedge F] &= 0, && (48)\\ \mathrm{d}\star F + [A \wedge \star F] &= 0. && (49) \end{aligned}$$

Here again equation (48) is *a priori* statement — the *Bianci identity*. Note that Maxwell's equations are a special case of Yang-Mills equations.

It is instructive to see what the Yang-Mills Lagrangian looks like in the discrete approximation. Replacing the Minkowski space $\mathcal{M}$ by a hypercubic lattice one can see that the discrete version of L_{YM} is proportional to $\sum_f \sigma\left(\gamma_f\right)$, where the summation is performed over all faces of a hypercubic constituent of the lattice;

$$\sigma = 2\dim\rho\left(\Gamma\right) - \chi\left(\rho\left(\Gamma\right)\right) - \chi\left(\rho^{\dagger}\left(\Gamma\right)\right);$$

where $\rho\left(\Gamma\right)$ and $\rho^{\dagger}\left(\Gamma\right)$ are fundamental representation of Γ and its dual, respectively; χ is the character; γ_f is the gauge group holonomy around the face f.

The Yang-Mills theory uses Hodge operation converting k-forms to $(n-k)$-forms in n-dimensional space *with metric* $g_{\mu\nu}$. In topological applications so-called *BF theory* plays an important role since it does not require a metric. In this theory, an additional dynamical field B is introduced. The Lie algebra valued $(n-2)$-form B and the 2-form F are combined into the Lagrangian $L_{BF} = \mathrm{Tr}\left[B \wedge F\right]$.

6.2. Quantum Behavior and Gauge Connection

The Aharonov–Bohm effect (Fig. 9) is one of the most remarkable illustrations of interplay between quantum behavior and gauge connection. Charged particles moving through the region containing perfectly shielded thin solenoid produce different interference patterns on a screen depending on whether the solenoid is turned on or off. There is no electromagnetic force acting on the particles, but working solenoid produces $\mathrm{U}(1)$-connection adding or subtracting phases of the particles and thus changing the interference pattern.

In the discrete time Feynman's path amplitude [26] is decomposed into the product of elements of the fundamental representation $\rho\left(\Gamma\right) = \mathrm{U}(1)$ of the circle, i.e., of the Lie group $\Gamma = \mathrm{S}^1 = \mathbb{R}/\mathbb{Z}$:

$$A_{\mathrm{U}(1)} = \exp\left(iS\right) = \exp\left(i\int Ldt\right) \longrightarrow \mathrm{e}^{iL_{0,1}}\ldots\mathrm{e}^{iL_{t-1,t}}\ldots\mathrm{e}^{iL_{T-1,T}}. \quad (50)$$

By the notation $L_{t-1,t}$ we emphasize that the Lagrangian is in fact a function defined on pairs of points (graph edges) — this is compatible with physics where the typical Lagrangians are depend on the *first order* derivatives. Thus we can interpret the expression $\pi(t-1,t) = e^{iL_{t-1,t}} \in \rho\left(\Gamma\right) = \mathrm{U}(1)$ as $\mathrm{U}(1)$-parallel transport.

A natural generalization of this is to suppose that:

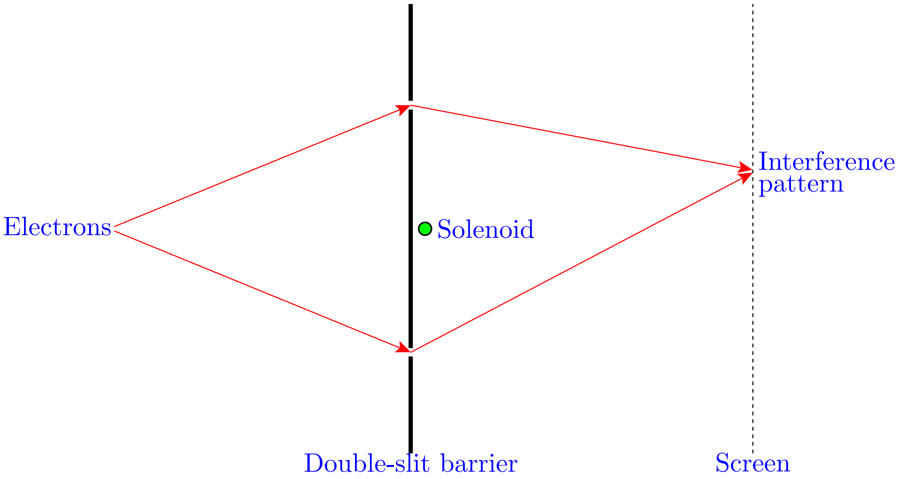


Figure 9. Aharonov–Bohm effect. Magnetic flux is confined within the perfectly shielded solenoid; interference pattern is shifted in spite of absence of electromagnetic forces acting on the particles.

- group Γ may differ from S^1,
- dimension of unitary representation $\rho\left(\Gamma\right)$ may differ from 1.

So let us replace expression (50) for Feynman's path amplitude by the following parallel transport along the path

$$A_{\rho(\Gamma)} = \rho\left(\alpha_{T,T-1}\right)\ldots\rho\left(\alpha_{t,t-1}\right)\ldots\rho\left(\alpha_{1,0}\right). \qquad (51)$$

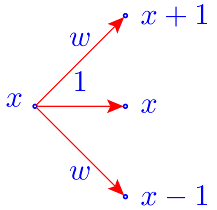
Here $\alpha_{t,t-1}$ are elements of some group Γ — we shall call it *quantizing group* — and ρ is an unitary representation of Γ. Note that in (50) the order of factors is not important due to commutativity of $\mathrm{U}(1)$. But in (51) we must use the reverse[15] order for consistency with the temporal ordering of non-commutative operators. For discrete and especially finite systems it is natural to take a finite group as the quantizing group, in this case all manipulations — in contrast to the standard quantization — remain within the framework of constructive discrete mathematics requiring no more than the ring of *algebraic integers* (and sometimes the quotient field of this ring). On the other hand, the standard quantization can be approximated by taking 1-dimensional representations of large enough finite groups.

6.2.1. Illustrative Example Inspired by Free Particle.

In quantum mechanics — as is clear from the *never vanishing* expression $\exp\left(\frac{i}{\hbar}S\right)$ for the path amplitude — transitions from one to any other state are possible in principle. But we shall consider computationally more tractable models with restricted sets of possible transitions.

[15] This awkwardness stems from the tradition to write operator actions on the left (cf. footnote 4 on page 122).

Let us consider quantization of a free particle moving in one dimension. Such a particle is described by the Lagrangian $L = \frac{m\dot{x}^2}{2}$. Assuming that there are only transitions to the closest points in the discretized space we come to the following rule for the one-time-step transition amplitudes



$$e^{\frac{i}{\hbar}\frac{m\{(x+1)-x\}^2}{2}} = e^{i\frac{m}{2\hbar}}$$

$$e^{\frac{i}{\hbar}\frac{m(x-x)^2}{2}} = \mathbf{1}$$

$$e^{\frac{i}{\hbar}\frac{m\{(x-1)-x\}^2}{2}} = e^{i\frac{m}{2\hbar}}.$$

That is, we have evolution rule as an $\mathrm{U}(1)$-valued function R defined on pairs of points (graph edges). Symbolically:

$$\begin{aligned} R\,(x \to x) &= 1 \in \mathrm{U}(1), \\ R\,(x \to x-1) = R\,(x \to x+1) &= w = e^{i\frac{m}{2\hbar}} \in \mathrm{U}(1). \end{aligned} \qquad (52)$$

Now let us assume that w in (52) is an element of some representation of a finite group: $w = \rho\,(\alpha)\,,\ \alpha \in \Gamma = \{\gamma_1 = \mathbf{1}, \ldots, \gamma_M\}$. Rearranging *multinomial coefficients* — *trinomial* in this concrete case — it is not difficult to write the sum amplitude over all paths from the space-time point $(0,0)$ to the point (x,t)

$$A_x^t\,(w) = \sum_{\tau=0}^{t} \frac{\tau!}{\left(\frac{\tau-x}{2}\right)!\,\left(\frac{\tau+x}{2}\right)!} \times \frac{t!}{\tau!\,(t-\tau)!}\, w^{\tau}. \qquad (53)$$

Note that x must lie in the limits determined by t: $x \in [-t, t]$.

One of the most expressive peculiarities of quantum-mechanical behavior is the *destructive interference* — cancellation of non-zero amplitudes attached to different paths converging to the same point. By construction, the sum of amplitudes in our model is a function $A(w)$ depending on distribution of sources of the particles, their initial phases, gauge fields acting along the paths, restrictions — like, e.g., "slits" — imposed on possible paths, etc. In the case of 1-dimensional representation the function $A(w)$ is a polynomial with algebraic integer coefficients and w is a root of unity. Thus the condition for destructive interference can be expressed by the system of polynomial equations: $A(w) = 0$ and $w^M = 1$. For concreteness let us consider the cyclic group $\Gamma = \mathbb{Z}_M = \{\gamma_1, \cdots, \gamma_k, \cdots, \gamma_M\}$. Any of its M irreducible representations takes the form $\rho\,(\gamma_k) = w^{k-1}$, where w is one of the Mth roots of unity. For simplicity let w be the *primitive root*: $w = \mathrm{e}^{2\pi i/M}$.

Fig. 10 shows all possible transitions (with their amplitudes) from the point x in three time steps. We see that the polynomial $A_{\pm 1}^3 = 3w + 3w^3 = 3w\left(w^2+1\right)$ contains the *cyclotomic polynomial* $\Phi_4(w) = w^2+1$ as a factor. The smallest group associated to $\Phi_4(w)$ — and hence providing the destructive interference — is $\mathbb{Z}_4$, which we shall consider as quantizing group for the model.

Fig. 11 shows interference patterns — normalized squared amplitudes ("probabilities") — from two sources placed in the positions $x = -4$ and $x = 4$ for 20 time steps. The upper and lower graph show interference pattern when sources are in the same ($\Delta\phi = 0$) and in the opposite ($\Delta\phi = \pi$) phases, respectively.

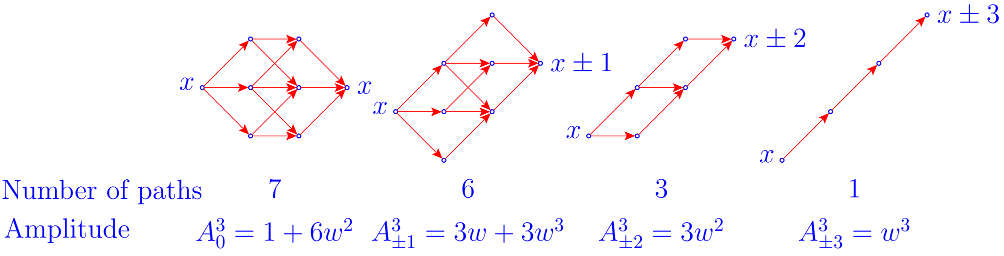


Figure 10. Amplitudes for all possible paths in three time steps.

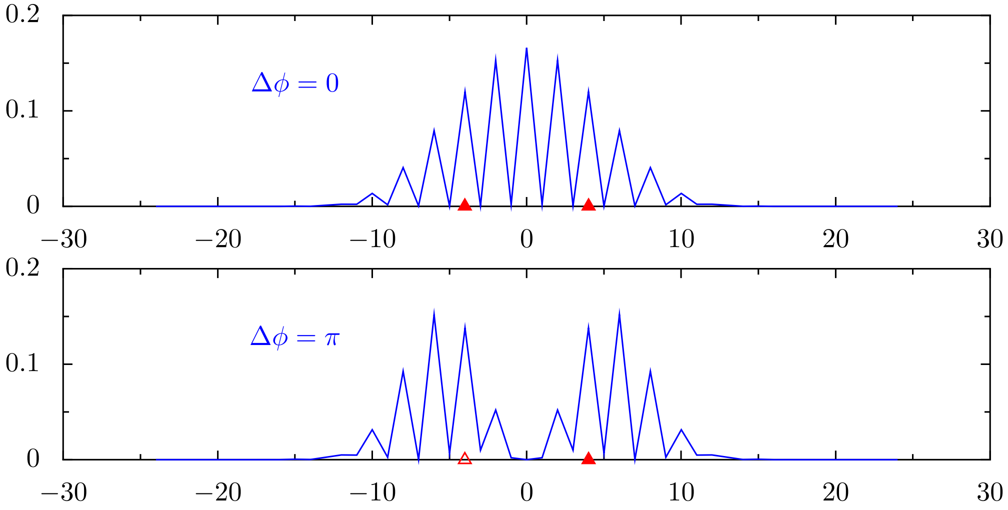


Figure 11. Group $\mathbb{Z}_4$. Interference from two sources at points -4 and 4. Number of time steps $T=20$. Phase differences $\Delta\phi=\phi_4-\phi_{-4}$ between sources are 0 and π.

6.2.2. Local Quantum Models on Regular Graphs

The above model — with quantum transitions allowed only within the neighborhood of a vertex of a 1-dimensional lattice — can easily be generalized to arbitrary regular graph. Our definition of *local quantum model on k-valent graph* uncludes the following:

1. *Space* $X=\{x_1,\cdots,x_{N_\mathrm{X}}\}$ is a $k-$valent graph.

2. *Set of local transitions* $E_i=\{e_{0,i},e_{1,i},\cdots,e_{k,i}\}$ is the set of k adjacent to the vertex x_i edges $e_{m,i}=(x_i\rightarrow x_{m,i})$ completed by the edge $e_{0,i}=(x_i\rightarrow x_i)$.

3. We assume that the *space symmetry* group $\mathrm{G}=\mathrm{Aut}\left(\mathrm{X}\right)$ acts transitively on the set $\{E_1,\cdots,E_{N_\mathrm{X}}\}$.

4. $\mathrm{G}_{\mathrm{loc}}=\mathrm{Stab}_{\mathrm{G}}\left(x_i\right)\leq\mathrm{G}$ is the *stabilizer* of x_i.

5. $\Omega_i=\{\omega_{0,i},\omega_{1,i},\cdots,\omega_{h,i}\}$ is the *set of orbits* of $\mathrm{G}_{\mathrm{loc}}$ on E_i.

6. *Quantizing group* Γ is a finite group: $\Gamma=\{\gamma_1,\cdots,\gamma_M\}$.

7. *Evolution rule* R is a function on E_i with values in some representation $\rho\left(\Gamma\right)$. The rule R prescribes $\rho\left(\Gamma\right)$-weights to the one-time-step transitions from x_i to elements of the neighborhood of x_i. From the symmetry considerations R must be a function on orbits from Ω_i, i.e., $R\left(e_{m,i}g\right)=R\left(e_{m,i}\right)$ for $g\in\mathrm{G}_{\mathrm{loc}}$.

To illustrate these constructions, let us consider the local quantum model on the graph of *buckyball* (see detailed consideration of this graph at page 126). Here the space $X=\{x_1,\cdots,x_{60}\}$ has the symmetry group $\mathrm{G}=\mathrm{Aut}\left(\mathrm{X}\right)=\mathbb{Z}_2\times\mathrm{Alt}(5)$. The set of local transitions takes the form $E_i=\{e_{0,i},\ e_{1,i},\ e_{2,i},\ e_{3,i}\}$, where

$$\begin{aligned}
e_{0,i} &= \left(x_i\rightarrow x_i\right),\\
e_{1,i} &= \left(x_i\rightarrow x_{1,i}\right),\\
e_{2,i} &= \left(x_i\rightarrow x_{2,i}\right),\\
e_{3,i} &= \left(x_i\rightarrow x_{3,i}\right).
\end{aligned}$$

The stabilizer of x_i is $\mathrm{G}_{\mathrm{loc}}=\mathrm{Stab}_{\mathrm{G}}\left(x_i\right)=\mathbb{Z}_2$. The set of orbits of $\mathrm{G}_{\mathrm{loc}}$ on E_i contains 3 orbits:

$$\Omega_i=\left\{\omega_{0,i}=\left\{e_{0,i}\right\},\omega_{1,i}=\left\{e_{1,i},e_{2,i}\right\},\omega_{2,i}=\left\{e_{3,i}\right\}\right\},$$

i.e., the stabilizer does not move the edges $\left(x_i\rightarrow x_i\right)$ and $\left(x_i\rightarrow x_{3,i}\right)$ and swaps $\left(x_i\rightarrow x_{1,i}\right)$ and $\left(x_i\rightarrow x_{2,i}\right)$.

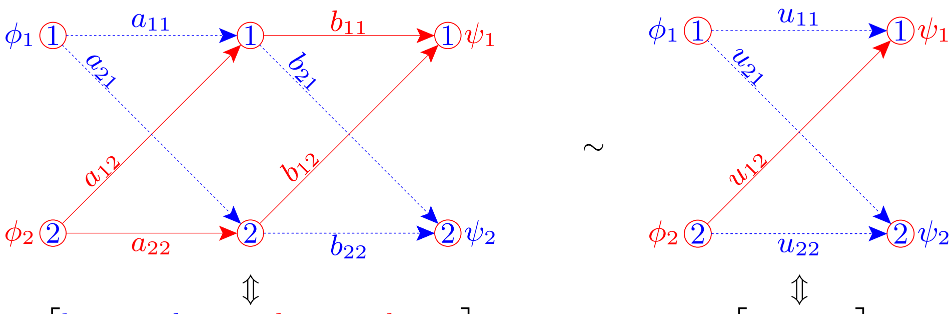
The evolution rule takes the form:

$$\begin{aligned}
R\left(x_i\rightarrow x_i\right) &= \rho\left(\alpha_0\right),\\
R\left(x_i\rightarrow x_{1,i}\right)=R\left(x_i\rightarrow x_{2,i}\right) &= \rho\left(\alpha_1\right),\\
R\left(x_i\rightarrow x_{3,i}\right) &= \rho\left(\alpha_2\right),
\end{aligned}$$

where $\alpha_0,\alpha_1,\alpha_2\in\Gamma$. If we take a 1-dimensional representation and move α_0 — using gauge invariance — to the identity element of Γ, we see that the rule R depends on two elements $v=\rho\left(\alpha_1\right)$ and $w=\rho\left(\alpha_2\right)$. Thus the amplitudes in the quantum model on the buckyball take the form $A(v,w)$ depending on two roots of unity. To search nontrivial quantizing groups one should check (by, e.g., Gröbner basis computation) compatibility of the system of polynomial equations $A(v,w)=\Phi_i(v)=\Phi_j(w)=0$, where $\Phi_i(v)$ and $\Phi_j(w)$ are cyclotomic polynomials.

6.3. General Discussion of Quantization in Finite Systems

As is well known, Feynman's approach is equivalent to the traditional matrix formulation of quantum mechanics, where the time evolution $\left|\psi_0\right\rangle\rightarrow\left|\psi_T\right\rangle$ of a system from the initial state vector to the final is described by the evolution matrix U: $\left|\psi_T\right\rangle=U\left|\psi_0\right\rangle$. The evolution matrix can be represented as the product of matrices corresponding to the single time steps: $U=U_{T\leftarrow T-1}\cdots U_{t\leftarrow t-1}\cdots U_{1\leftarrow 0}$. In fact, Feynman's quantization — i.e., the rules "multiply subsequent events" and "sum up alternative histories" — is simply a rephrasing of matrix multiplication. This is clear from the below illustration presenting two-time-step evolution of a two-state system (single *qubit*) in both Feynman's and matrix forms — the general case of many time steps and many states can easily be obtained (by induction for example).



$$BA = \begin{bmatrix} b_{11}a_{11} + b_{12}a_{21} & b_{11}a_{12} + b_{12}a_{22} \\ b_{21}a_{11} + b_{22}a_{21} & b_{21}a_{12} + b_{22}a_{22} \end{bmatrix} \qquad \sim \qquad U = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix}$$

We see that in accordance with Feynman's rules the transition from, e.g., ϕ_2 to ψ_1 is determined by the expression $b_{11}a_{12} + b_{12}a_{22}$. But this is just the element u_{12} of the matrix product $U = BA$ performing evolution $|\psi\rangle = U\,|\phi\rangle$, where $|\phi\rangle = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix}$ and $|\psi\rangle = \begin{bmatrix} \psi_1 \\ \psi_2 \end{bmatrix}$.

Of course, such reduction of sums over histories to matrices is applicable to the case of transitions along paths being gauge connections as in (51). In this case matrix elements of an $N_{\mathrm{X}} \times N_{\mathrm{X}}$ evolution matrix U are themselves matrices from the representation $\rho\left(\Gamma\right)$. We can ignore this particular block structure of the matrix and consider U as an $N \times N$ matrix over the field $\mathbb{C}$, where $N = N_{\mathrm{X}} \times \dim \rho\left(\Gamma\right)$.

In quantum mechanics, the evolution matrices U are unitary operators acting in Hilbert spaces of state vectors (called also "wave functions", "amplitudes" etc.). Quantum mechanical particles are associated with unitary representations of some groups. According to their dimensions, these representations are called "singlets", "doublets", etc. Multidimensional representations describe the *spin*. A quantum mechanical experiment is reduced to comparison of the system state vector $|\psi\rangle$ with some sample state vector $|\phi\rangle$. According to the Born rule, the probability to observe coincidence of the states is equal to $\left|\langle\phi|\psi\rangle\right|^2$, where $\langle\cdot|\cdot\rangle$ is the inner product in the Hilbert space. To see what these constructions may look like in the constructive finite background, let us assume that evolution operators are elements of a representation of a finite group.

6.3.1. Permutations and Linear Representations

Having a finite group $G = \{e_1 = \mathbf{1}, \ldots, e_m\}$, we can easily describe all its transitive actions on finite sets [27]. Any such set $\Omega = \{\omega_1, \ldots, \omega_n\}$ is in one-to-one correspondence with the *right* (or *left*) *cosets* of some subgroup $H \leq G$, i.e., $\Omega \cong H \setminus G$ (or G/H) is the *homogeneous space* (or *G-space*). Action of G on Ω is *faithful* if the subgroup H does not contain *normal sugroups* of G. We can write actions in terms of permutations

$$\pi(g) = \begin{pmatrix} \omega_i \\ \omega_i g \end{pmatrix} \sim \begin{pmatrix} Ha \\ Hag \end{pmatrix}, \qquad g, a \in G, \quad i = 1, \ldots, n.$$

Maximum transitive set Ω is the group itself, i.e., in the above construction $H = \{\mathbf{1}\}$. The action on $\Omega = G$ is called *regular* and can be represented by permutations

$$\Pi(g) = \begin{pmatrix} e_i \\ e_i g \end{pmatrix}, \quad i = 1, \ldots, m. \tag{54}$$

To introduce "numerical" ("statistical") description, let us assume that ω_i's are basis elements of a linear vector space $\mathcal{H}$ over a field F

$$\mathcal{H} = \mathrm{Span}\left(\omega_1, \cdots, \omega_n\right), \tag{55}$$

i.e., we prescribe F-valued "weights" to the elements $\omega_i \in \Omega$. Then we can write permutations in the matrix form:

$$\pi(g) \rightarrow \rho(g) = \left[\rho(g)_{ij}\right], \quad \text{where } \rho(g)_{ij} = \delta_{\omega_i g, \omega_j};\ \ i, j = 1, \ldots, n; \tag{56}$$

$$\delta_{\alpha,\beta} \equiv \begin{cases} 1, & \text{if } \alpha = \beta, \\ 0, & \text{if } \alpha \neq \beta \end{cases} \quad \text{for } \alpha, \beta \in \Omega.$$

The so defined function ρ is called a *permutation representation*. The matrix form of (54)

$$\Pi(g) \rightarrow \mathrm{P}(g) = \left[\mathrm{P}(g)_{ij}\right], \ \ \mathrm{P}(g)_{ij} = \delta_{e_i g, e_j}, \ \ i, j = 1, \ldots, m \tag{57}$$

is called the *regular representation*. It is assumed that F is an algebraically closed field — usually the field of complex numbers $\mathbb{C}$. But in the case of finite groups the quotient field of the ring $\mathbb{A}$ of *algebraic integers*[16] [12] is sufficient for all reasonable purposes — $\mathbb{A}$ is a *constructive* subset of $\mathbb{C}$.

Let us recall some relevant background information about linear representations of finite groups [28].

1. *Any linear* representation of a finite group G is *unitary* since there is always an unique invariant inner product $\langle\cdot|\cdot\rangle$ making any space of representation $\mathcal{H}$ into a Hilbert space.

2. *All possible* irreducible unitary representations of the group G are contained in the regular representation (57). More specifically, all matrices (57) can simultaneously be reduced by some unitary transformation S to the form

$$\mathrm{S}^{-1}\mathrm{P}(g)\mathrm{S} = \begin{bmatrix} \Delta_1(g) & & & & & & \\ & d_2\left\{\begin{matrix}\Delta_2(g) & & \\ & \ddots & \\ & & \Delta_2(g)\end{matrix}\right. & & & \\ & & \ddots & & \\ & & & d_r\left\{\begin{matrix}\Delta_r(g) & & \\ & \ddots & \\ & & \Delta_r(g)\end{matrix}\right. \end{bmatrix}. \tag{58}$$

Here r is the number of different irreducible representations Δ_j of the group G. This number coincides with the number of *conjugacy classes*[17] in G. The number d_j is

[16] The ring of algebraic integers consists of the roots of *monic* polynomials with integer coefficients. A polynomial is called *monic* if its leading coefficient is unit.

[17] The jth conjugacy class $C_j \subseteq G$ consists of all group elements of the form $g^{-1}c_j g$, where $c_j \in C_j$ is some (arbitrary) representative of the class, $g \in G$, $j = 1, \ldots, r$.

simultaneously the dimension of Δ_j and its multiplicity in the regular representation, so it is obvious that $d_1^2 + d_2^2 + \cdots + d_r^2 = |G| = m$. It can be proved also that any d_j divides the number of elements of G: $d_j \mid m$.

3. Any irreducible representation Δ_j is determined uniquely (up to isomorphism) by its *character* χ_j. The character is a function on G defined as $\chi_j(a) = \mathrm{Tr}\Delta_j(a),\ a \in G$. The character is a *central* or *class* function, i.e., it is constant on the conjugacy classes: $\chi_j(a) = \chi_j(g^{-1}ag),\ a, g \in G$. Any class function $\varphi(a)$ on G is a linear combination of the characters $\chi_1, \ldots, \chi_r$.

4. All values of χ_j and eigenvalues of Δ_j are elements of the ring $\mathbb{A}$ of algebraic integers, moreover the eigenvalues are *roots of unity.*

5. A convenient form of describing all irreducible representation of a finite group G is the *character table*. The columns of this table correspond to the conjugacy classes of G while its rows correspond to the characters χ_j of the inequivalent irreducible representations of G.

	$\mathbf{1}$	c_2	$\cdots$	c_r
χ_1	1	1	$\cdots$	1
χ_2	$\chi_2(c_1)$	$\chi_2(c_2)$	$\cdots$	$\chi_2(c_r)$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
χ_r	$\chi_r(c_1)$	$\chi_r(c_2)$	$\cdots$	$\chi_r(c_r)$

The jth column is indicated by a representative $c_j \in C_j$ of the jth conjugacy class C_j. Conventionally we take $c_1 = \mathbf{1}$ and χ_1 to be the *trivial character* corresponding to the trivial 1-dimensional representation.

6.3.2. Interpretation of Quantum Description in Finite Background

Let us discuss sketchy (more detailed presentation see in [29]) constructive approach to the interpretation of quantum description.

Summarizing the above, we see that dynamics of finite quantum model of any type is reduced ultimately to a single finite-dimensional unitary $k \times k$ matrix U describing transitions between initial and final vectors in some k-dimensional Hilbert space $\mathcal{H}_k$. In the finite background the matrix U is an element of unitary representation Δ of a finite group G, i.e., the number of all possible evolutions is equal to $m = |G|$. We shall assume, as is accepted in quantum mechanics, that Δ is direct sum of irreducible representations Δ_j from (58). The decomposition of the Hilbert space into irreducible components is an important part of the mathematical formulation of quantum mechanics. Such dependence on the choice of the basis in the Hilbert space may seem unusual for a physical theory. But, in fact, a basis in which the Hilbet space is reduced — we shall call such a basis *quantum basis* — simply reflects the structure of underlying symmetry group.

We can construct a G-space $\Omega = \{\omega_1, \ldots, \omega_n\}$[18] in such a way that its permutation representation (56) contains Δ as subrepresentation (obviously $n \geq k$). That is, the space

[18] In the case that Δ is reducible representation, the set Ω may be intransitive union of transitive G-spaces.

$\mathcal{H}_k$ is subspace of the Hilbert space $\mathcal{H}_n$ of the permutation representation. We shall call the basis $\{\omega_1, \ldots, \omega_n\}$ in the space $\mathcal{H}_n$ the *permutation basis*. Transitions from the permutation to quantum basis for matrices $\widetilde{U}$ and vectors $\left|\widetilde{\psi}\right\rangle \in \mathcal{H}_n$ are given by the formulas

$$\begin{aligned} \widetilde{U}_q &= S^{-1}\widetilde{U}_p S, && (59) \\ \left|\widetilde{\psi}_q\right\rangle &= S^{-1}\left|\widetilde{\psi}_p\right\rangle. && (60) \end{aligned}$$

Now we can embed any evolution U with the matrix Δ in the space $\mathcal{H}_k$ into the evolution $\widetilde{U}$ in the space $\mathcal{H}_n$. In the quantum basis the matrix of $\widetilde{U}$ takes the form

$$\widetilde{U}_q = \begin{bmatrix} \Delta & 0 \\ 0 & A \end{bmatrix}, \tag{61}$$

where A is an $(n-k)\times(n-k)$ matrix. Due to the form of (61) the evolution U described by Δ is completely independent of the components of vectors of $\mathcal{H}_n$ related to A. The "hidden variables" that can come from the additional components describe degrees of freedom reflecting indistinguishability of ω_i's lying in the same group orbit. The evolution $\widetilde{U}$ is simply a permutation of ω_i's and can not manifest anything quantum.

Illustration. A quantum model with the group $\mathrm{Sym}(3)$**.** The group $G = \mathrm{Sym}(3)$ is the group of all permutations of three objects. This is the smallest non-commutative group. Its 6 elements form the following 3 conjugate classes $C_1 = \{\mathbf{1} = ()\}$, $C_2 = \{a_1 = (12), a_2 = (23), a_3 = (13)\}$, $C_3 = \{b_1 = (123), b_2 = (132)\}$. We used here the *cyclic notation* for permutations. The group has three nonequivalent irreducible representations described by the character table

	$\mathbf{1}$	a_i	b_i
χ_1	1	1	1
χ_2	1	-1	1
χ_3	2	0	-1

.

Let us take for example the 2-dimensional representation Δ with the character χ_3. The representation is given by the following set of 2×2 matrices:

$$\Delta(\mathbf{1}) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\Delta(a_1) = \begin{bmatrix} 0 & \mathrm{e}^{-2\pi i/3} \\ \mathrm{e}^{2\pi i/3} & 0 \end{bmatrix}, \quad \Delta(a_2) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \Delta(a_3) = \begin{bmatrix} 0 & \mathrm{e}^{2\pi i/3} \\ \mathrm{e}^{-2\pi i/3} & 0 \end{bmatrix},$$

$$\Delta(b_1) = \begin{bmatrix} \mathrm{e}^{2\pi i/3} & 0 \\ 0 & \mathrm{e}^{-2\pi i/3} \end{bmatrix}, \quad \Delta(b_2) = \begin{bmatrix} \mathrm{e}^{-2\pi i/3} & 0 \\ 0 & \mathrm{e}^{2\pi i/3} \end{bmatrix}.$$

The regular permutation representation of $\mathrm{Sym}(3)$ is 6-dimensional. But 3-dimensional faithful permutation representation induced by the action on the homogeneous space $\mathrm{Sym}(2)\setminus\mathrm{Sym}(3) \cong \Omega = \{\omega_1, \omega_2, \omega_3\}$ also contains Δ. Since any permutation representation contains trivial 1-dimension subrepresentation, the only possible choice of the addition A is the representation corresponding to the first row of the above character table. Thus, for $\widetilde{U}_q$ we have

$$\widetilde{U}_q(\mathbf{1}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\widetilde{U}_q(a_1) = \begin{bmatrix} 0 & \mathrm{e}^{-2\pi i/3} & 0 \\ \mathrm{e}^{2\pi i/3} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \widetilde{U}_q(a_2) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \widetilde{U}_q(a_3) = \begin{bmatrix} 0 & \mathrm{e}^{2\pi i/3} & 0 \\ \mathrm{e}^{-2\pi i/3} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\widetilde{U}_q(b_1) = \begin{bmatrix} \mathrm{e}^{2\pi i/3} & 0 & 0 \\ 0 & \mathrm{e}^{-2\pi i/3} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \widetilde{U}_q(b_2) = \begin{bmatrix} \mathrm{e}^{-2\pi i/3} & 0 & 0 \\ 0 & \mathrm{e}^{2\pi i/3} & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

In the permutation basis we have

$$\widetilde{U}_p(\mathbf{1}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \widetilde{U}_p(a_1) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \widetilde{U}_p(a_2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$

$$\widetilde{U}_p(a_3) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \widetilde{U}_p(b_1) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad \widetilde{U}_p(b_2) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

The most general unitary matrix of transition from the permutation to the quantum basis takes the form

$$S = \frac{\mathrm{e}^{\alpha i}}{\sqrt{3}} \begin{bmatrix} 1 & 1 & \mathrm{e}^{\beta i} \\ \mathrm{e}^{2\pi i/3} & \mathrm{e}^{-2\pi i/3} & \mathrm{e}^{\beta i} \\ \mathrm{e}^{-2\pi i/3} & \mathrm{e}^{2\pi i/3} & \mathrm{e}^{\beta i} \end{bmatrix}, \quad \text{where } \alpha, \beta \text{ are arbitrary real parameters.} \tag{62}$$

Any quantum evolution of the form $\left|\psi\right\rangle = U\left|\phi\right\rangle$, where $\left|\phi\right\rangle = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix}$ and $\left|\psi\right\rangle = \begin{bmatrix} \psi_1 \\ \psi_2 \end{bmatrix}$, and U is one of the matrices Δ; can be extended to the evolution $\left|\widetilde{\psi}_q\right\rangle = \widetilde{U}_q\left|\widetilde{\phi}_q\right\rangle$, where $\left|\widetilde{\phi}_q\right\rangle = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}$ and $\left|\widetilde{\psi}_q\right\rangle = \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \end{bmatrix}$, ϕ_3 is arbitrary additional component. Then, applying the transformation S, we come to the classical evolution with the matrix $\widetilde{U}_p = S\widetilde{U}_qS^{-1}$ which simply permutes the components of the initial vector

$$\left|\widetilde{\phi}_p\right\rangle = S\left|\widetilde{\phi}_q\right\rangle = \frac{\mathrm{e}^{\alpha i}}{\sqrt{3}} \begin{bmatrix} \phi_1 + \phi_2 + \mathrm{e}^{\beta i}\phi_3 \\ \mathrm{e}^{2\pi i/3}\phi_1 + \mathrm{e}^{-2\pi i/3}\phi_2 + \mathrm{e}^{\beta i}\phi_3 \\ \mathrm{e}^{-2\pi i/3}\phi_1 + \mathrm{e}^{2\pi i/3}\phi_2 + \mathrm{e}^{\beta i}\phi_3 \end{bmatrix}$$

without performing any algebraic manipulations with the components.

7. Conclusion

In this chapter we discuss the general concept of discrete dynamical system and its specialization involving underlying space structures. We apply various constructive approaches to study discrete and finite dynamical systems.

We construct a family of groups unifying space and internal symmetries in a natural way. This construction generalizes the standard direct and wreath products.

We introduce the concept of a system of discrete relations on an abstract simplicial complex. This system can be treated as a natural generalization of cellular automata or as a set-theoretical analog of systems of polynomial equations.

We developed and implemented algorithms for analyzing compatibility of systems of discrete relations and for constructing canonical decompositions of discrete relations.

Applying the technique described above to some cellular automata — a particular case of discrete relations — we obtained a number of results. The most interesting among them, in our opinion, is the demonstration of how the presence of non-trivial proper consequences may determine the global behavior of an automaton.

We suggest an algorithmic approach — based on discrete symmetry analysis and implemented in C — for construction and investigation of discrete dynamical models — deterministic, mesoscopic and quantum. We hope that our approach can be used in various practical applications, such as, for example, simulation of nanostructures with nontrivial symmetry properties.

We demonstrate that soliton-like moving structures — like “spaceships” in cellular automata, “traveling waves” in mathematical physics and “generalized coherent states” in quantum physics — arise inevitably in deterministic dynamical systems whose symmetry group splits the set of states into finite number of group orbits.

We formulate the gauge principle in the form most suitable for discrete and finite systems. We also propose a method — based on introduction of unitary gauge connection of a special kind — for quantizing discrete systems and construct simple models for studying properties of suggested quantization.

We show that if unitary operators describing dynamics of finite quantum system form finite group, then the system can be embedded into a classical system with a simple behavior. We hope that discrete and finite background allowing comprehensive study may lead to deeper understanding of the quantum behavior and its connection with symmetries of systems.

To study more complicated models we are developing C programs based on computer algebra and computational group theory methods.

Acknowledgments

This work was supported by the grant 10-01-00200 from the Russian Foundation for Basic Research and by the grant 3810.2010.2 from the Ministry of Education and Science of the Russian Federation.

References

[1] Verlinde, E.P. (2010). *On the Origin of Gravity and the Laws of Newton,* arXiv:1001.0785

[2] t Hooft, G. (1993). *Dimensional reduction in quantum gravity,* Utrecht preprint THU-93/26; gr-qc/9310006.

[3] Kornyak, V.V. (2005). On Compatibility of Discrete Relations, *Lect. Notes Comp. Sci.* **3718**, Springer-Verlag Berlin Heidelberg, 272–284.

[4] Kornyak, V.V. (2006). Discrete Relations On Abstract Simplicial Complexes, *Programming and Computer Software*. **32**, No 2, 84–89.

[5] Kornyak, V.V. (2006). Cellular Automata with Symmetric Local Rules, *Lect. Notes Comp. Sci.* **4194**, Springer-Verlag Berlin Heidelberg, 240–250.

[6] Kornyak, V.V. (2008). Discrete Dynamical Systems with Symmetries: Computer Analysis, *Programming and Computer Software*. **34**, No 2, 84–94.

[7] Kornyak, V.V. (2009). Discrete Dynamics: Gauge Invariance and Quantization, *Lect. Notes Comp. Sci.* **5743**, Springer-Verlag Berlin Heidelberg, 180–194.

[8] Holt, D.F.; Eick, B.; O'Brien, E. A. (2005). *Handbook of Computational Group Theory*. Chapman & Hall/CRC Press

[9] Seiberg, N. (2006). Emergent Spacetime. Rapporteur talk at the 23rd Solvay Conference in Physics, December, 2005. hep-th/0601234.

[10] McKay, B.D. (1981). Practical Graph Isomporphism. *Congressus Numerantium* **30**, 45–87, http://cs.anu.edu.au/ bdm/nauty/PGI

[11] Klein, F. (1884). *Vorlesungen über das Ikosaeder*. Leipzig: Teubner. Translated to Russian under the title *Lektsii ob ikosaedre i reshenii uravnenii pyatoi stepeni*, Moscow: Nauka, 1989.

[12] Kirillov, A.A. (1976). *Elements of the Theory of Representations.* Springer-Verlag, Berlin-New York.

[13] Lidl, R.; Niederreiter, H. (1983). *Finite Fields;* Reading, Mass.: Addison-Wesley.

[14] Hilton, P.; Wayle, S. (1960). *Homology Theory: An Introduction to Algebraic Topology,* Cambrige Univ. Press.

[15] http://psoup.math.wisc.edu/mcell/rullex_life.html

[16] Wolfram, S. (2002). *A New Kind of Science;* Wolfram Media, Inc

[17] http://atlas.wolfram.com/01/01/

[18] ’t Hooft, G. (1999). Quantum Gravity as a Dissipative Deterministic System. SPIN-1999/07, gr-qc/9903084; *Class. Quant. Grav.* **16**, 3263 (1999); also published in: *Fundamental Interactions: from symmetries to black holes* (Conference held on the occasion of the “Eméritat” of François Englert, 24-27 March 1999, ed. by J.-M. Frère et al, Univ. Libre de Bruxelles, Belgium, 221–240.

[19] ’t Hooft, G. (2006). The mathematical basis for deterministic quantum mechanics. ITP-UU-06/14, SPIN-06/12, quant-ph/0604008, 1–17.

[20] Imry, Y. (2002). *Introduction to Mesoscopic Physics (Mesoscopic Physics and Nanotechnology, 2).* Oxford University Press, USA, 256 p.

[21] Gross, D.H.E. (2001). *Microcanonical thermodynamics: Phase transitions in “Small” Systems.* World Scientific, Singapore, 269 p.

[22] Gross, D.H.E. (2004). A New Thermodynamics from Nuclei to Stars. *Entropy*, **6**, 158-179

[23] Gross, D.H.E.; Votyakov, E.V. (2000). Phase Transitions in “Small” Systems. *Eur. Phys. J. B*, **15**, 115-126.

[24] Ispolatov, I.; Cohen, E. G. D. (2001). On First-order Phase Transitions in Microcanonical and Canonical Non-extensive Systems, *Physica A* **295**, 475–487.

[25] Oeckl, R. (2005). *Discrete Gauge Theory (From Lattices to TQPT)*. Imperial College Press, London.

[26] Feynman R.P.; Hibbs A.R. (1965). *Quantum Mechanics and Path Integrals.* McGraw-Hill.

[27] Marshall Hall, Jr. (1959). *The Theory of Groups.* Macmillan Co., New York.

[28] Serre, J.-P. (1977). *Linear Representations of Finite Groups.* Springer-Verlag.

[29] Kornyak, V.V. (2010). *Finite Quantum Models: Constructive Approach to Description of Quantum Behavior*, 19 p. http://arxiv.org/abs/1010.3370

In: Cellular Automata
Editor: Thomas M. Li, pp.165-209

ISBN: 978-1-61761-592-4

Chapter 8

REVERSIBILITY OF CELLULAR AUTOMATA

Atsushi Nobe*[*]*and Fumitaka Yura
Department of Mathematics, Faculty of Education, Chiba University
School of Systems Information Science, Future University-Hakodate

Abstract

We establish a on-to-one correspondence between the configurations in the Wolfram cellular automaton, which is abbreviated to the WCA, and the paths in the de Bruijn quiver. Extending the correspondence to that between the associative algebra whose underlying vector space is generated by the configurations in the WCA and the path algebra of the de Bruijn quiver, we obtain the global transition of the associative algebra associated with the WCA. Thus we translate the problem concerning reversibility of the WCA into that concerning surjectivity of the endomorphism on the associative algebra. We then show that the induced problem concerning the endomorphism can be solved in terms of the adjacency matrix of the WCA, which is defined from that of the de Bruijn quiver through the one-to-one correspondence. Indeed, we give a necessary and sufficient condition for reversibility of the WCA. By virtue of the necessary and sufficient condition, we classify all 16 reversible rules in the ECA imposing periodic boundary conditions.

PACS 02.10.Ox, 02.30.Ik, 05.65.+b, 87.17.-d

Keywords: cellular automata, path algebras of quivers, discrete integrable systems.

1. Introduction

For the last several decades there has been increasing interest in the study of discrete dynamical systems including cellular automata, which take discrete values in discrete time steps, from the viewpoints of solvability and integrability. Such studies on cellular automata, *e.g.*, traffic flow models [32, 33, 14, 15, 16, 17] and the Toda type cellular automata [25, 28, 29, 3, 30], have had great success. In this context, one of the most interesting and well-studied systems is a family of cellular automata called the box-ball system

[*] E-mail address: nobe@faculty.chiba-u.jp

[41, 40, 36, 6, 46, 45, 22] originated from the soliton cellular automaton introduced by Takahashi and Satsuma in 1990 [37]. Each member of the box-ball system is a kind of filter type cellular automaton and has solitonical nature, *i.e.*, in its time evolution, solitarily propagating waves behave like particles when they collide each other. A remarkable and important feature of the box-ball system is that each member can be directly connected with the partial differential equation called a soliton equation such as the KdV equation, the KP equation, and the Toda equation in terms of a procedure called the ultradiscretization [40, 10, 11]. Here "directly connected" means that the evolution equation of a member of the box-ball system can be obtained from a soliton equation by applying the ultradiscretization procedure, which consists of two processes of changing of variables upon introduction of a parameter and taking a limit of the parameter. Moreover, through the ultradiscretization procedure, almost all integrable properties such as soliton solutions and sufficiently many conserved quantities are completely preserved [40, 41, 11, 9, 12]. Imposing periodic boundary conditions to the box-ball system, such property is still preserved [46, 45, 18, 19, 20, 21, 23]. Therefore each member of the box-ball systems is considered to be an ultimate discretization of an integrable system and called a integrable cellular automaton. Nowadays it is well known that the box-ball system is deeply connected with various mathematical structures such as the crystals of quantum algebra [6, 4], the Riemann hypothesis [39], tropical algebraic curves [7, 8, 12], and so on.

Although there exist various integrable dynamical systems other than the soliton equations, as far as the authors know, integrable cellular automata are scarcely known except for the box-ball system. Of course, the definition of integrability of cellular automata has not been fixed yet and to define it is too much for the authors; however, we may call a cellular automaton which has the general solution for arbitrary initial condition with appropriate boundary conditions to be integrable. To search generic cellular automata for integrable one is too hard to be made because there exist infinitely many cellular automata around us. Therefore it is natural for us to restrict ourselves to a certain subclass of cellular automata. Noticing that, in general, reversibility of a dynamical system with respect to the time evolution is a necessary condition for its integrability, it is natural to consider that an integrable cellular automaton must be reversible, *i.e.*, the inverse time evolution is uniquely determined. Therefore we expect to find integrable cellular automata among reversible ones. In order to carry this out, we must establish a criterion to decide whether a given cellular automaton is reversible or not. Note that, for our purpose, an algorithm to examine reversibility of a given cellular automaton under a certain initial condition is not sufficient. Since integrability of a dynamical system does not depend on its system size or initial condition, we essentially want to decide reversibility of a family consisting of infinitely many cellular automata under arbitrary initial condition. Thus we have been developed an algebraic method to examine reversibility of cellular automata [34, 35].

In this chapter, we give a procedure which bijectively associates a family of cellular automata to a family of quivers (oriented graphs), and by using the one-to-one correspondence we extend the time evolution of a family of cellular automata to that of an associative algebra generated by the configurations of the cellular automata. The procedure we discuss here was firstly introduced by the authors in 2004 [34]. Then the process to decide reversibility of cellular automata can be reduced to a purely algebraic process in the associative algebra; some of them are easily carried out and we can examine reversibility of a family consisting of infinitely many cellular automata. The procedure we give here is a powerful tool to study cellular automata, however, it is applicable only to one-dimensional cellular automata, because the correspondence between a cellular automaton and a quiver depends essentially on “the order” of the local configurations of the cellular automaton. Nevertheless, the authors believe that our method is worth to study. Because although the updating rule of one-dimensional cellular automaton, which is composed of regular array of finite-valued cells updated locally in discrete time steps, is quite simple, a cellular automaton in general shows complicated behavior [42, 43, 44]. As we will see later, boundary conditions play an important role for reversibility of cellular automata. Among several choices of boundary conditions we choose the periodic one because it can be considered to be the most natural one. Since time evolution of a cellular automaton with periodic boundary conditions can be regarded as a mapping from a finite set into itself, if the mapping is surjective then the rule is reversible. A reversible cellular automaton preserves all information of the initial configurations in any time step, hence its reversibility suggests existence of conserved quantities [35]. We give an example of the time evolution of a reversible cellular automaton in Figure 1.

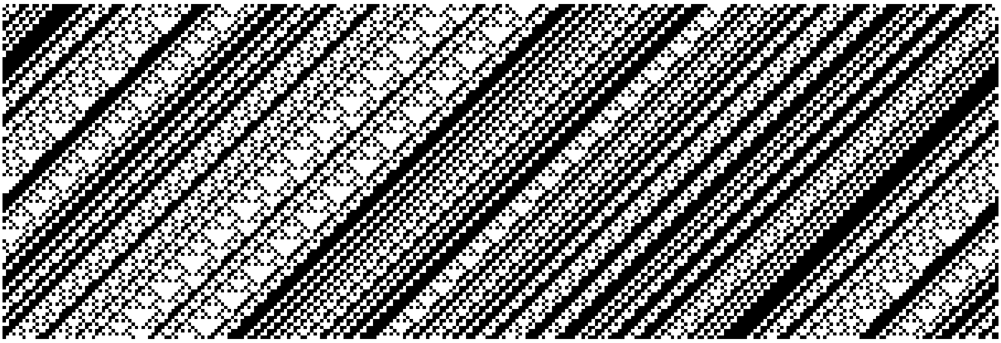


Figure 1. Time evolution of the reversible elementary cellular automaton referred to the rule 154 (see example 12) with periodic boundary conditions. In the initial configuration the values of cells are chosen randomly 0 or 1. Cells with value 1 are shown by black and those with value 0 by blanks. The configurations at successive time steps are shown on successive lines from top to bottom.

This chapter is organized as follows. In section 2., we introduce a notion of quivers and the path algebras of the quivers. In particular, we define a family of quivers called the de Bruijn quiver and study it precisely. In this process, we introduce the adjacency matrix of the de Bruijn quiver, and show that the many properties of the de Bruijn quiver can be computed by using the adjacency matrix. We then define cellular automata and study it in section 3. Especially, we establish a one-to-one correspondence between a family of cellular automata called the Wolfram cellular automaton and the de Bruijn quiver. In section 4., we study reversibility of the Wolfram cellular automaton in terms of the one-to-one correspondence to the de Bruijn quiver. We then give a necessary and sufficient condition for reversibility of the Wolfram cellular automaton. We also give two necessary conditions for reversibility of the Wolfram cellular automaton, which are useful to examine reversibility of a given rule. We classify all reversible rules in a subclass of the Wolfram cellular automaton called the elementary cellular automaton in section 5. Section 6. is devoted to concluding remarks.

2. Quivers

A quiver $Q = (Q_0, Q_1, s, t)$ is a quadruple consisting of two sets Q_0, Q_1 and two maps $s, t : Q_1 \to Q_0$. Elements in Q_0 and in Q_1 are called vertices and arrows, respectively. The maps s and t associate to each arrow $\alpha \in Q_1$ its source $s(\alpha) \in Q_0$ and target $t(\alpha) \in Q_0$, respectively. We usually denote an arrow $\alpha \in Q_1$ of source $a = s(\alpha) \in Q_0$ and target $b = t(\alpha) \in Q_0$ by

$$(a|\alpha|b)\,.$$

Although a quiver is nothing but an oriented graph, we prefer using the term quiver to graph. Because all graphs we treat here are oriented and the orientation of graphs is essential in the correspondence to cellular automata. Moreover, the path algebra of a quiver, which will be defined soon (see definition 1), plays an important role in the study of reversibility of cellular automata.

Let $a, b \in Q_0$. A path of length $n \geq 1$ with source a and target b is a sequence

$$(a|\alpha_1\alpha_2\cdots\alpha_n|b)\,,$$

where $\alpha_k \in Q_1$ for all $1 \leq k \leq n$, and we have $s(\alpha_1) = a$, $t(\alpha_k) = s(\alpha_{k+1})$ for $1 \leq k \leq n-1$ and $t(\alpha_n) = b$. We denote by Q_n the set of all paths in Q of length n. We also associate with each vertex $a \in Q_0$ a path of length $n = 0$, called the stationary path at a, and denoted by

$$\varepsilon_a = (a||a)\,.$$

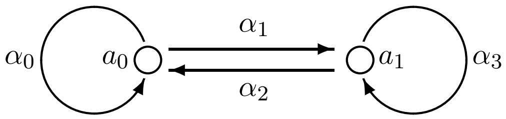
A path of length $n \geq 1$ is called a cycle whenever its source and target coincide. A cycle of length 1 is called a loop.

Definition 1. (see for example [2]) Let Q be a quiver. Also let K be an algebraically closed field. The path algebra KQ of Q is the K-algebra whose underlying K-vector space has as its basis the set of all paths $(a|\alpha_1\alpha_2\cdots\alpha_n|b)$ of length $n \geq 0$ in Q and such that the product of two basis vectors $(a|\alpha_1\alpha_2\cdots\alpha_n|b)$ and $(c|\beta_1\beta_2\cdots\beta_k|d)$ of KQ is defined by

$$(a|\alpha_1\alpha_2\cdots\alpha_n|b)\,(c|\beta_1\beta_2\cdots\beta_k|d) = \delta_{bc}\,(a|\alpha_1\alpha_2\cdots\alpha_n\beta_1\beta_2\cdots\beta_k|d)\,,$$

where δ_{bc} denotes the Kronecker delta. The product of basis elements is then extended to arbitrary elements of KQ by distributivity.

Example 1. Let Q be the quiver



consisting of two vertices a_0, a_1 and four arrows α_i ($i = 0, 1, 2, 3$), two of which α_0, α_3 are loops. The defining basis of the path algebra KQ is the set of all words on $\{\alpha_0, \alpha_1, \alpha_2, \alpha_3\}$ with the identification

$$\alpha_0\alpha_2 = \alpha_0\alpha_3 = \alpha_1\alpha_0 = \alpha_1\alpha_1 = \alpha_2\alpha_2 = \alpha_2\alpha_3 = \alpha_3\alpha_0 = \alpha_3\alpha_1 = 0.$$

Therefore, KQ is isomorphic to the quotient of the free associative algebra in four non-commuting indeterminates $K\langle t_0, t_1, t_2, t_3\rangle$ in terms of the two-sided ideal $J = \langle t_0t_2, t_0t_3, t_1t_0, t_1t_1, t_2t_2, t_2t_3, t_3t_0, t_3t_1\rangle$

$$KQ \simeq K\langle t_0, t_1, t_2, t_3\rangle / J,$$

the K-linear isomorphism being a map such that

$$\varepsilon_0 + \varepsilon_1 \mapsto 1 \qquad \text{and} \qquad \alpha_i \mapsto t_i \quad \text{for } i = 0, 1, 2, 3,$$

where we abbreviate ε_{a_i} to ε_i, and hereafter we often use such abbreviation.

A quiver is said to be finite if both Q_0 and Q_1 are finite sets. To each arrow $(a|\alpha|b)$ in a quiver Q, we associate a formal reverse $(b|\alpha^{-1}|a)$ with the source $s(\alpha^{-1}) = b$ and the target $t(\alpha^{-1}) = a$. A walk of length $n \geq 1$ from a to b in Q is a sequence $w = \alpha_1^{\pi_1}\alpha_2^{\pi_2} \cdots \alpha_n^{\pi_n}$ with $\pi_i \in \{-1, 1\}$, $s(\alpha_1^{\pi_1}) = a$, $t(\alpha_n^{\pi_n}) = b$ and $t(\alpha_i^{\pi_i}) = s(\alpha_{i+1}^{\pi_{i+1}})$ for $i = 1, 2, \ldots, n$. A quiver is said to be connected if each pair of vertices is joined by a walk.

Definition 2. Let Q be a finite and connected quiver. The two-sided ideal of the path algebra KQ generated by the arrows in Q is called the arrow ideal of KQ and is denoted by R_Q.

There is a direct sum decomposition

$$KQ = KQ_0 \oplus R_Q,$$
$$R_Q = \bigoplus_{l \geq 1} KQ_n$$

of K-vector spaces KQ and R_Q, where KQ_n ($n = 0, 1, \ldots$) is the subspace of KQ generated by the set Q_n of all paths of length n. For each $n \geq 1$ we have

$$R_Q^n = \bigoplus_{m \geq n} KQ_m.$$

Therefore R_Q^n is the ideal of KQ generated by the set of all paths of length not less than n. Consequently, we have the K-vector space isomorphism $R_Q^n / R_Q^{n+1} \simeq KQ_n$.

2.1. De Bruijn Quiver

Now we introduce a two-parameter family of quivers called the de Bruijn quiver [26]. Let $l, r \geq 1$ be natural numbers. Each member of the de Bruijn quiver, denoted by $\mathcal{B}(l, r) = (Q_0, Q_1, s, t)$, can be defined as follows. The set Q_0 of the vertices in $\mathcal{B}(l, r)$ contains r^l elements $a_0, a_1, \cdots, a_{r^l-1}$, and the set Q_1 of the arrows in $\mathcal{B}(l, r)$ contains r^{l+1} elements

$\alpha_0, \alpha_1, \cdots, \alpha_{r^{l+1}-1}$. An arrow α_i connects the vertices a_j to a_k, *i.e.*, $s(\alpha_i) = a_j$ and $t(\alpha_i) = a_k$, if the following holds

$$j = \left\lfloor \frac{i}{r} \right\rfloor \qquad \text{and} \qquad k \equiv i \pmod{r^l}, \tag{1}$$

where $\lfloor \ \rfloor : \mathbb{Q} \to \mathbb{Z}$ is the floor function.

It immediately follows several properties of the de Bruijn quiver.

Lemma 1. Each member $\mathcal{B}(l, r)$ of the de Bruijn quiver has the following properties:

1. The vertex a_j is the source of the r arrows $\alpha_{r\cdot j}$, $\alpha_{r\cdot j+1}$, $\alpha_{r\cdot j+2}$, $\cdots$, $\alpha_{r\cdot j+r-1}$ for $j = 0, 1, \ldots, r^l - 1$.
2. The vertex a_k is the target of the r arrows α_k, α_{r^l+k}, α_{2r^l+k}, $\cdots$, $\alpha_{(r-1)r^l+k}$ for $k = 0, 1, \ldots, r^l - 1$.
3. Every successive pair (α_i, α_j) of arrows in a path satisfies
$$j \equiv r\cdot i, r\cdot i + 1, \cdots, r\cdot i + r - 1 \pmod{r^{l+1}}.$$
4. The vertex $a_{\frac{r^l-1}{r-1}i}$ is the source (or equivalently the target) of the loop $\alpha_{\frac{r^{l+1}-1}{r-1}i}$ for $i = 0, 1, \ldots, r - 1$.
5. There exists a unique path of length l which connects arbitrary two vertices in $\mathcal{B}(l, r)$.

(Proof) The statements 1 and 2 are a direct consequence from the definition of the de Bruijn quiver. Indeed, assume a_j to be the source of an arrow α_i. Then, from (1), we have

$$i = r\cdot j, r\cdot j + 1, \cdots, r\cdot j + r - 1.$$

Similarly, assume a_k to be the target of an arrow α_i. Then, from (1), we have

$$i = k, r^l + k, 2r^l + k, \cdots, (r-1)r^l + k.$$

3. Assume (α_i, α_j) to be a successive pair of arrows in a path in $\mathcal{B}(l, r)$. Then there exist $0 \le k \le r^l - 1$ which satisfies

$$t(\alpha_i) = s(\alpha_j) = a_k \in Q_0,$$

where Q_0 is the set of all vertices in $\mathcal{B}(l, r)$. From the statements 1 and 2, there exist $0 \le s \le r - 1$ and $0 \le t \le r - 1$ such that

$$j = r\cdot k + s \qquad \text{and} \qquad i = t\cdot r^l + k.$$

Eliminating k, we obtain

$$j = r\cdot\left(i - t\cdot r^l\right) + s \equiv r\cdot i + s \pmod{r^{l+1}}.$$

4. Assume α_j to be a loop in $\mathcal{B}(l, r)$. Then the pair (α_j, α_j) of arrows is successive in a path. Therefore, from the statement 3, we have

$$j \equiv r \cdot j + i \quad \text{mod } (r^{l+1})$$

for a certain $0 \leq i \leq r - 1$. Put $j = (r^{l+1} - 1)i/(r - 1)$. Then we have

$$\begin{aligned} r \cdot j + i &= r\frac{r^{l+1} - 1}{r - 1}i + i \\ &= \left(1 + r + r^2 + \cdots + r^l + r^{l+1}\right) i \\ &\equiv \left(1 + r + r^2 + \cdots + r^l\right) i \quad (\text{mod } r^{l+1}) \\ &= \frac{r^{l+1} - 1}{r - 1}i = j. \end{aligned}$$

From the statement 1 (or equivalently 2), the source (or equivalently the target) a_k of the loop α_j satisfies

$$j = \frac{r^{l+1} - 1}{r - 1}i = r \cdot k + i.$$

Thus we obtain

$$k = \frac{r^l - 1}{r - 1}i.$$

5. Let us consider the path of length l

$$\left(a_j | \alpha_{i_0} \alpha_{i_1} \cdots \alpha_{i_{l-2}} \alpha_{i_{l-1}} | a_k\right)$$

whose source and target are a_j and a_k, respectively. From the statement 3 we have

$$\begin{aligned} &i_{l-1} \equiv r \cdot i_{l-2} + s_{l-1} \quad (\text{mod } r^{l+1}), \\ &i_{l-2} \equiv r \cdot i_{l-3} + s_{l-2} \quad (\text{mod } r^{l+1}), \\ &\cdots \\ &i_1 \equiv r \cdot i_0 + s_1 \quad (\text{mod } r^{l+1}), \end{aligned}$$

where $0 \leq s_1, \cdots, s_{l-2}, s_{l-1} \leq r - 1$. Therefore we inductively have

$$i_{l-1} \equiv r^{l-1} \cdot i_0 + \sum_{m=1}^{l-1} r^{m-1} s_{l-m} \quad (\text{mod } r^{l+1}).$$

Since $k \equiv i_{l-1}$ (mod r^l), we have

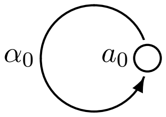
$$k \equiv r^{l-1} \cdot i_0 + \sum_{m=1}^{l-1} r^{m-1} s_{l-m} \quad (\text{mod } r^l).$$

Note that, by appropriate choice of $s_1, \cdots, s_{l-2}, s_{l-1}$, the term $\sum_{m=1}^{l-1} r^{m-1} s_{l-m}$ stands for an arbitrary integer from 0 to $r^{l-1} - 1$. By the statement 1, for a given j, we can choose i_0 from among the following r integers

$$r \cdot j, r \cdot j + 1, \cdots, r \cdot j + r - 1.$$

Therefore, k stands for an arbitrary integer from 0 to $r^l - 1$. Thus, for a given $0 \leq j \leq r^l - 1$, there exists a unique k among $0, 1, \ldots, r^l - 1$. This is nothing but arbitrary two vertices to be connected with by a unique path of length l in $\mathcal{B}(l, r)$. □

Example 2. Assume $r = 1$. Then we obtain $\mathcal{B}(l, 1) = (Q_0 = \{a_0\}, Q_1 = \{\alpha_0\}, s, t)$. The de Bruijn quiver $\mathcal{B}(l, 1)$ can be realized as a planner graph as follows.



The defining basis of the path algebra $K\mathcal{B}(l, 1)$ is $\{\varepsilon_0, \alpha_0, \alpha_0^2, \cdots, \alpha_0^m, \cdots\}$. Therefore, $K\mathcal{B}(l, 1)$ is isomorphic to the polynomial algebra $K[t]$ in one indeterminate t, the isomorphism being induced by the K-linear map such that

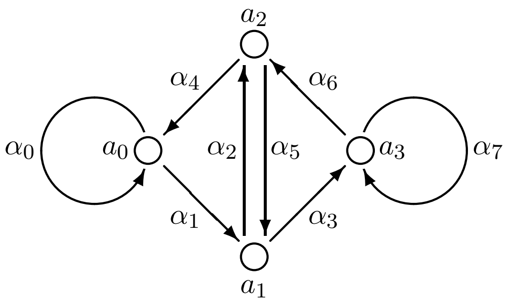
$$\varepsilon_0 \mapsto 1 \qquad \text{and} \qquad \alpha_0 \mapsto t.$$

Example 3. Assume $r = 2$. Then we obtain $\mathcal{B}(l, 2) = (Q_0 = \{a_0, a_1, \cdots, a_{2^l - 1}\}, Q_1 = \{\alpha_0, \alpha_1, \cdots, \alpha_{2^{l+1}-1}\}, s, t)$. The condition (1) can be summarized in the following table.

i	0	1	2	$\cdots$	$2^l - 2$	$2^l - 1$	2^l	$2^l + 1$	$\cdots$	$2^{l+1} - 1$
i (mod 2^l)	0	1	2	$\cdots$	$2^l - 2$	$2^l - 1$	0	1	$\cdots$	$2^l - 1$
$\uparrow$										
$\left\lfloor \frac{i}{2} \right\rfloor$	0	0	1	$\cdots$	$2^{l-1} - 1$	$2^{l-1} - 1$	2^{l-1}	2^{l-1}	$\cdots$	$2^l - 1$

This table says that the vertex whose index is in the bottom row is connected with the vertex whose index is in the second row in terms of the arrow whose index is in the top row.

The case when $l = 1$ is given in example 1. The quiver $\mathcal{B}(2, 2)$ can be drawn as follows.



The defining basis of the path algebra $K\mathcal{B}(l,2)$ is the set of all words on $\{\alpha_0, \alpha_1, \ldots, \alpha_{2^{l+1}-1}\}$ with the identification $\alpha_i\alpha_j = 0$ except for such (i,j) that

$$j \equiv 2i, 2i+1 \pmod{2^{l+1}} \qquad (i = 0, 1, \ldots, 2^{l+1}-1).$$

Thus $K\mathcal{B}(l,2)$ is isomorphic to the quotient of the free associative algebra $K\langle t_0, t_1, \cdots, t_{2^{l+1}-1}\rangle$ in 2^{l+1} non-commuting indeterminates $t_0, t_1, \cdots, t_{2^{l+1}-1}$ in terms of the two-sided ideal $J = \langle t_i t_j\rangle$, where i, j ranges over $\{0, 1, \ldots, 2^{l+1}-1\}$ except for $j \equiv 2i, 2i+1 \pmod{2^{l+1}}$

$$K\mathcal{B}(l,2) \simeq K\langle t_0, t_1, \cdots, t_{2^{l+1}-1}\rangle / J.$$

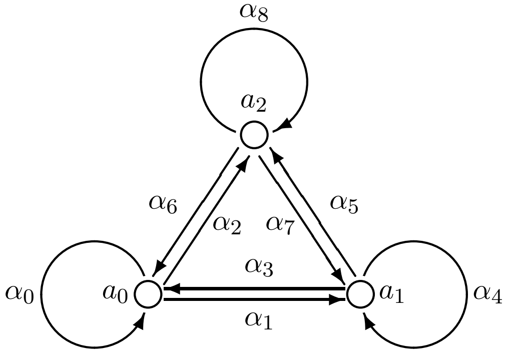
The K-linear isomorphism is a map such that

$$\sum_{j=0}^{2^l-1} \varepsilon_j \mapsto 1 \qquad \text{and} \qquad \alpha_i \mapsto t_i \quad \text{for } i = 0, 1, \ldots, 2^{l+1}-1.$$

Example 4. Assume $r = 3$. Then we obtain $\mathcal{B}(l,3) = (Q_0 = \{a_0, a_1, \cdots, a_{3^l-1}\}, Q_1 = \{\alpha_0, \alpha_1, \cdots, \alpha_{3^{l+1}-1}\}, s, t)$. The condition (1) can be summarized in the following table.

i	0	1	2	3	$\cdots$	3^l-2	3^l-1	3^l	3^l+1	3^l+2	$\cdots$	$\cdots$	$2\cdot 3^l-1$	$2\cdot 3^l$	$2\cdot 3^l+1$	$2\cdot 3^l+2$	$\cdots$	$3^{l+1}-1$
$i \pmod{3^l}$	0	1	2	3	$\cdots$	3^l-2	3^l-1	0	1	2	$\cdots$	$\cdots$	3^l-1	0	1	2	$\cdots$	3^l-1
$\uparrow$																		
$\left\lfloor \frac{i}{3} \right\rfloor$	0	0	0	1	$\cdots$	$3^{l-1}-1$	$3^{l-1}-1$	3^{l-1}	3^{l-1}	3^{l-1}	$\cdots$	$\cdots$	$2\cdot 3^{l-1}-1$	$2\cdot 3^{l-1}$	$2\cdot 3^{l-1}$	$2\cdot 3^{l-1}$	$\cdots$	3^l-1

As in the case of $\mathcal{B}(l,1)$ and $\mathcal{B}(l,2)$, we can draw the quiver $\mathcal{B}(l,3)$ as a planner graph for generic l. The simplest quiver $\mathcal{B}(1,3)$ can be drawn as follows.



The defining basis of the path algebra $K\mathcal{B}(l,3)$ is the set of all words on $\left\{\alpha_0,\alpha_1,\ldots,\alpha_{3^{l+1}-1}\right\}$ with the identification $\alpha_i\alpha_j=0$ except for such (i,j) that

$$j\equiv 3i,3i+1,3i+2 \pmod{3^{l+1}} \qquad (i=0,1,\ldots,3^{l+1}-1).$$

Thus $K\mathcal{B}(l,3)$ is isomorphic to the quotient of the free associative algebra $K\langle t_0,t_1,\cdots,t_{3^{l+1}-1}\rangle$ in 3^{l+1} non-commuting indeterminates $t_0,t_1,\cdots,t_{3^{l+1}-1}$ in terms of the two-sided ideal $J=\langle t_it_j\rangle$, where i,j ranges over $\{0,1,\ldots,3^{l+1}-1\}$ except for $j\equiv 3i,3i+1,3i+2 \pmod{3^{l+1}}$

$$K\mathcal{B}(l,3)\simeq K\langle t_0,t_1,\cdots,t_{3^{l+1}-1}\rangle/J.$$

The K-linear isomorphism is a map such that

$$\sum_{j=0}^{3^l-1}\varepsilon_j\mapsto 1 \qquad \text{and} \qquad \alpha_i\mapsto t_i \quad \text{for } i=0,1,\ldots,3^{l+1}-1.$$

2.2. Adjacency Matrices

The adjacency matrix of a finite quiver Q over the path algebra KQ is defined as follows.

Definition 3. (see for example [5]) Let $Q=(Q_0,Q_1,s,t)$ be a finite quiver. Assume that Q contains n vertices. The adjacency matrix of Q is the $n\times n$ matrix $\mathcal{M}=\mathcal{M}(Q)\in\mathrm{Mat}(n,KQ)$ whose entry μ_{ij} is

$$\mu_{ij}=\alpha+\beta+\cdots+\gamma\in KQ$$

if $s(\alpha)=s(\beta)=\cdots=s(\gamma)=a_i$ and $t(\alpha)=t(\beta)=\cdots=t(\gamma)=a_j$ for $a_i,a_j\in Q_0$ and $\alpha,\beta,\cdots,\gamma\in Q_1$; otherwise $\mu_{ij}=0$.

By definition, each entry of $\mathcal{M}$ is 0 or a linear combination of paths of length 1 in Q with coefficients 1, and every path of length 1 in Q appears in an entry of $\mathcal{M}$ just once. Remember that the product of elements in the path algebra KQ of the quiver Q is defined as a connection of the paths corresponding to the elements. Thus, for any integer $n>0$, each entry of the n-th power of $\mathcal{M}$ is 0 or a linear combination of paths of length n with coefficients 1, and every path of length n appears in an entry of $\mathcal{M}^n$ just once. In particular, the trace of $\mathcal{M}^n$ is the linear combination of all cycles of length n in Q with coefficients 1.

Note that the number of nonzero terms in the i-th row (resp. column) of the adjacency matrix of a quiver corresponds to that of arrows emanating from (resp. sinking into) the i-th vertex in the quiver. By definition and lemma 1, the adjacency matrix of the de Bruijn quiver can be characterized as follows.

Proposition 1. 1. Let μ_{jk} be the (j,k)-entry of the adjacency matrix of the de Bruijn quiver $\mathcal{B}(l,r)$. Then we have

$$\mu_{jk}=\begin{cases}\alpha_i & \text{if } j=\lfloor i/r\rfloor \text{ and } k\equiv i \pmod{r^l}\\ 0 & \text{otherwise.}\end{cases}$$

2. There exist exactly r nonzero entries in each row and column of the adjacency matrix of the de Bruijn quiver, respectively. □

By using the adjacency matrix, we can easily prove several properties of the de Bruijn quiver.

Proposition 2. Let $n \geq l$ be an integer. Then each member $\mathcal{B}(l,r)$ of the de Bruijn quiver has r^{n-l} paths of length n which connect arbitrary two vertices in $\mathcal{B}(l,r)$.

(Proof) There exist a unique path of length l connecting arbitrary two vertices in $\mathcal{B}(l,r)$ (see lemma 1). Therefor the l-th power $\mathcal{M}^l$ of the adjacency matrix of $\mathcal{B}(l,r)$ has no empty entry and every entry contains a monomial of degree l in $\alpha_0, \alpha_1, \cdots, \alpha_{r^{l+1}-1} \in Q_1$, where Q_1 is the set of all arrows in $\mathcal{B}(l,r)$.

Let us consider the product of the adjacency matrices:

$$\mathcal{M}^l \times \mathcal{M} = \mathcal{M}^{l+1}.$$

Since every column of $\mathcal{M}$ contains exactly r nonzero entries, every entry of $\mathcal{M}^{l+1}$ contains a homogeneous r-term polynomial of degree $l+1$ in $\alpha_0, \alpha_1, \cdots, \alpha_{r^{l+1}-1}$. Therefore there exist r paths of length $l+1$ connecting arbitrary two vertices in $\mathcal{B}(l,r)$. Induction on the power of $\mathcal{M}$ completes the proof. □

Corollary 1. Let $n \geq l$ be a natural number. Then each member $\mathcal{B}(l,r)$ of the de Bruijn quiver has r^n cycles of length n.

(Proof) By virtue of proposition 2, for every vertex in $\mathcal{B}(l,r)$, there exist r^{n-l} cycles of length n emanating from it; and there exist r^l vertices in $\mathcal{B}(l,r)$. Therefore there exist $r^{n-l} \times r^l = r^n$ cycles of length n in $\mathcal{B}(l,r)$. □

Corollary 2. Let $n \geq l$ be a natural number. Then each member $\mathcal{B}(l,r)$ of the de Bruijn quiver has r^{n+l} paths of length n.

(Proof) By virtue of proposition 2, there exist r^{n-l} paths of length n connecting arbitrary two vertices in $\mathcal{B}(l,r)$; and there exist possible $r^l \times r^l$ choices of two vertices in $\mathcal{B}(l,r)$. Therefore there exist $r^{n-l} \times r^{2l} = r^{n+l}$ paths of length n in $\mathcal{B}(l,r)$. □

We can obtain the explicit form of the trace of any power of the adjacency matrix of the de Bruijn quiver.

Proposition 3. Let $\mathcal{M}$ be the adjacency matrix of the de Bruijn quiver $\mathcal{B}(l,r)$. Then the trace of the n-th power of $\mathcal{M}$ is

$$\mathrm{tr}\,(\mathcal{M}^n) = \sum_{j_0, j_1, \cdots, j_{n-2} \in \{0,1,\ldots,r^{l+1}-1\}} \alpha_{j_0}\alpha_{j_1}\cdots\alpha_{j_{n-1}},$$

where j_i ranges over $\{0,1,\ldots,r^{l+1}-1\}$ with each successive pair (j_i, j_{i+1}) satisfying $j_{i+1} \equiv r \cdot j_i, r \cdot j_i + 1, \cdots, r \cdot j_i + r - 1 \pmod{r^{l+1}}$ for $i = 0, 1, \ldots, n-1$ and we assume $j_n = j_0$.

(Proof) By definition of the adjacency matrix, $\mathrm{tr}\,(\mathcal{M}^n)$ is the linear combination of all cycles of length n in $\mathcal{B}(l,r)$ with coefficient 1. Thus we obtain the following form

$$\mathrm{tr}\,(\mathcal{M}^n)=\sum_{j_0,j_1,\cdots,j_{n-2}\in\{0,1,\ldots,r^{l+1}-1\}}\alpha_{j_0}\alpha_{j_1}\cdots\alpha_{j_{n-1}}.$$

On the other hand, lemma 1 says that every successive pair (α_i,α_j) of arrows in a cycle in $\mathcal{B}(l,r)$ satisfies

$$j\equiv r\cdot i, r\cdot i+1,\cdots,r\cdot i+r-1 \pmod{r^{l+1}}.$$

Therefore each subscript j_i ranges over $\{0,1,\ldots,r^{l+1}-1\}$ with each successive pair (j_i,j_{i+1}) satisfying $j_{i+1}\equiv r\cdot j_i, r\cdot j_i+1,\cdots,r\cdot j_i+r-1$ (mod r^{l+1}). □

Example 5. Let Q be the de Bruijn quiver $\mathcal{B}(l,1)$. Then the adjacency matrix $\mathcal{M}$ of $\mathcal{B}(l,1)$ is the following 1×1 matrix

$$\mathcal{M}=(\alpha_0).$$

The trace of the n-th power of $\mathcal{M}$ is

$$\mathrm{tr}\,(\mathcal{M}^n)=\alpha_0^n.$$

Example 6. Let Q be the de Bruijn quiver $\mathcal{B}(l,2)$. Then the adjacency matrix $\mathcal{M}$ of $\mathcal{B}(l,2)$ is the following $2^l\times 2^l$ matrix

$$\mathcal{M}=\begin{pmatrix}
\alpha_0 & \alpha_1 & 0 & 0 & \cdots & \cdots & 0 & 0\\
0 & 0 & \alpha_2 & \alpha_3 & \cdots & \cdots & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots\\
0 & 0 & 0 & 0 & \cdots & \cdots & \alpha_{2^l-2} & \alpha_{2^l-1}\\
\alpha_{2^l} & \alpha_{2^l+1} & 0 & 0 & \cdots & \cdots & 0 & 0\\
0 & 0 & \alpha_{2^l+2} & \alpha_{2^l+3} & \cdots & \cdots & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots\\
0 & 0 & 0 & 0 & \cdots & \cdots & \alpha_{2^{l+1}-2} & \alpha_{2^{l+1}-1}
\end{pmatrix}.$$

The trace of the n-th power of $\mathcal{M}$ is

$$\mathrm{tr}\,(\mathcal{M}^n)=\sum_{j_0,j_1,\cdots,j_{n-2}\in\{0,1,\ldots,2^{l+1}-1\}}\alpha_{j_0}\alpha_{j_1}\cdots\alpha_{j_{n-1}},$$

where j_i ranges over $\{0,1,\ldots,2^{l+1}-1\}$ with each successive pair (j_i,j_{i+1}) satisfying $j_{i+1}\equiv 2j_i, 2j_i+1$ (mod 2^{l+1}) for $i=0,1,\ldots,n-1$.

Example 7. Let Q be the de Bruijn quiver $\mathcal{B}(l,3)$. Then the adjacency matrix $\mathcal{M}$ of $\mathcal{B}(l,3)$ is the following $3^l \times 3^l$ matrix

$$\mathcal{M} = \begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_3 & \cdots & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & \alpha_{3^l-3} & \alpha_{3^l-3} & \alpha_{3^l-1} \\ \alpha_{3^l} & \alpha_{3^l+1} & \alpha_{3^l+2} & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_{3^l+3} & \cdots & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & \alpha_{2\cdot 3^l-3} & \alpha_{2\cdot 3^l-2} & \alpha_{2\cdot 3^l-1} \\ \alpha_{2\cdot 3^l} & \alpha_{2\cdot 3^l+1} & \alpha_{2\cdot 3^l+2} & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_{2\cdot 3^l+3} & \cdots & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & \alpha_{3^{l+1}-3} & \alpha_{3^{l+1}-2} & \alpha_{3^{l+1}-1} \end{pmatrix}.$$

The trace of the n-th power of $\mathcal{M}$ is

$$\operatorname{tr}(\mathcal{M}^n) = \sum_{j_0, j_1, \cdots, j_{n-2} \in \{0,1,\ldots,3^{l+1}-1\}} \alpha_{j_0}\alpha_{j_1}\cdots\alpha_{j_{n-1}},$$

where j_i ranges over $\{0,1,\ldots,3^{l+1}-1\}$ with each successive pair (j_i, j_{i+1}) satisfying $j_{i+1} \equiv 3j_i, 3j_i+1, 3j_i+2$ (mod 3^{l+1}) for $i = 0,1,\ldots,n-1$.

3. Cellular Automata

Let $C = (E, \pi, B)$ be a fiber bundle

$$\pi : E \to B,$$

where E is the total space, B is the base space, and $\pi : E \to B$ is the projection. If the base space B and all the fibers $F_x = \pi^{-1}(x)$ over $x \in B$ are discrete spaces the fiber bundle C is called a cell. A section $s : B \to E$ of C

$$\pi \circ s(x) = x \qquad \text{for } x \in B$$

is called a global configuration of the cell C.

Let $\{U_\lambda\}_{\lambda\in\Lambda}$ be an open covering of B and $s_\lambda : U_\lambda \to E$ be a local section on U_λ:

$$\pi \circ s_\lambda(x) = x \qquad \text{for } x \in U_\lambda.$$

We call s_λ a local configuration. Let $\Gamma(E)$ be the set of all global configurations of the cell C. Also let $\Gamma_\lambda(E)$ be the set of all local configurations on U_λ.

Let $\sigma : B \to \{U_\lambda\}_{\lambda\in\Lambda}$, $\sigma(x) = U_\lambda \ni x$, where U_λ is a coordinate neighborhood of x. In terms of σ, we fix a coordinate neighborhood U_λ of $x \in B$. A continuous map $T_\lambda : \Gamma_\lambda(E) \to F_x = \pi^{-1}(x)$ is called the local transition of the cell C on the coordinate neighborhood $\sigma(x) = U_\lambda$ of x. The induced map $T : \Gamma(E) \to \Gamma(E)$ from the local transitions $\{T_\lambda\}_{\lambda\in\Lambda}$ is called the global transition of the cell C. If all local transitions are isomorphic then the global transition T is said to be homogeneous.

Definition 4. A cellular automaton is a quadruple $A = (C, \{U_\lambda\}_{\lambda\in\Lambda}, \sigma, T)$ consisting of a cell $C = (E, \pi, B)$, open covering $\{U_\lambda\}_{\lambda\in\Lambda}$ of B, a map $\sigma : B \to \{U_\lambda\}_{\lambda\in\Lambda}$, and the global transition $T : \Gamma(E) \to \Gamma(E)$ induced from the local transitions on U_λ for $\lambda \in \Lambda$. If the global transition is homogeneous then A is called a homogeneous cellular automaton.

Example 8. Assume $E = \mathbb{Z} \times \{0, 1\}$, $B = \mathbb{Z}$, and $\pi(x, f) = x$ for $x \in B$ and $f \in \{0, 1\}$. Then the cell $C = (E, \pi, B)$ is a trivial bundle. Take the coordinate neighborhood $\sigma(x) = U_\lambda$ of $x \in B$

$$U_\lambda = \{x - 1, x, x + 1\}.$$

Then we have

$$s_\lambda(x-1) = (x-1, f_{x-1}), \quad s_\lambda(x) = (x, f_x), \quad s_\lambda(x+1) = (x+1, f_{x+1}),$$

where $f_{x-1}, f_x, f_{x+1} \in \{0, 1\}$.

If we set

$$T_\lambda(s_\lambda) = f_{x-1} + f_x + f_{x+1} = \bar{f}_x \in F_x \qquad \text{for } x \in U_\lambda,$$

where the addition is considered modulo 2, then the induced map T is the global transition of C. Therefore, since T_λ is independent of λ, $A = (C, \{U_\lambda\}_{\lambda\in\Lambda}, \sigma, T)$ is a homogeneous cellular automaton and is nothing but Wolfram's elementary cellular automaton referred to the rule 150.

If we assume the cell C and the open covering U_λ to be as in example 8, we obtain all 256 rules in the elementary cellular automaton, which is often abbreviated to the ECA, by appropriate choices of the local transition. Indeed, let

$$T_\lambda(s_\lambda) = \delta(f_{x-1}, f_x, f_{x+1}) = \bar{f}_x \in F_x,$$

then, since $f_{x-1}, f_x, f_{x+1}, \bar{f}_x \in \{0, 1\}$, there exist 256 possibilities of choosing $\delta : \{0, 1\}^3 \to \{0, 1\}$. If we assign the number

$$\rho = \sum_{i_0, i_1, i_2 \in \{0,1\}} \delta(i_0, i_1, i_2) 2^{2^2 i_0 + 2^1 i_1 + 2^0 i_2}$$

to the local transition T_λ given by δ, we obtain Wolfram's ECA referred to the rule ρ.

Choosing a higher dimensional base space B of a cell C, we obtain a cellular automaton of dimension $n > 1$.

Example 9. Assume $E = \mathbb{Z}^2 \times \{0, 1\}$, $B = \mathbb{Z}^2$, and $\pi((x, y), f) = (x, y)$ for $(x, y) \in B$ and $f \in \{0, 1\}$. Then the cell $C = (E, \pi, B)$ is a trivial bundle. Take the coordinate neighborhood $\sigma(x, y) = U_\lambda$ of $(x, y) \in B$

$$U_\lambda = \{(x + i, y + j) \in B \mid i, j \in \{-1, 0, 1\}\}.$$

Then we have

$$s_\lambda(x+i, y+j) = \big((x+i, y+j), f_{(x+i,y+j)}\big) \qquad \text{for } i, j \in \{-1, 0, 1\},$$

where $f_{(x+i,y+j)} \in \{0,1\}$.

If we set

$$T_\lambda(s_\lambda) = R(f_{(x+i,y+j)}) = \bar{f}_{(x,y)} \in F_{(x,y)} \qquad \text{for } (x,y) \in U_\lambda,$$

where

$$R(f_{(i,j)}) = \begin{cases} 1 & \text{if } f_0 = 1, \displaystyle\sum_{i,j=-1}^{1} f_{(x+i,y+j)} = 3, 4 \text{ or } f_0 = 0, \displaystyle\sum_{i,j=-1}^{1} f_{(x+i,y+j)} = 3 \\ 0 & \text{otherwise} \end{cases}$$

then the induced map T is the global transition of C. Therefore, since T_λ is independent of λ, $A = (C, \{U_\lambda\}_{\lambda\in\Lambda}, \sigma, T)$ is a homogeneous cellular automaton and is nothing but Conway's game of life.

3.1. Wolfram Cellular Automaton

Now we consider a family of cellular automata called the Wolfram cellular automaton, which is abbreviated to the WCA hereafter [42, 43, 44]. Let $l, r \geq 1$ be natural numbers. Each member of the WCA, denoted by $\mathcal{W}(l,r) = (C(r), \{U_\lambda(l)\}_{\lambda\in\Lambda}, \sigma, T)$, can be defined as follows. The cell $C(r)$ of $\mathcal{W}(l,r)$ is a trivial bundle given by the total space $E = \mathbb{Z} \times \{0, 1, \ldots, r-1\}$, the base space $B = \mathbb{Z}$, and the projection $\pi(x, f) = x$ for $f \in \{0, 1, \ldots, r-1\}$. The coordinate neighborhood $\sigma(x)$ of $x \in B$ is given as follows

$$\sigma(x) = U_\lambda(l) = \{x + e(0), x + e(1), \cdots, x + e(l)\},$$

where $e : \mathbb{Z} \to \mathbb{Z}$ is defined by

$$e(i) = -\left\lfloor \frac{l}{2} \right\rfloor + i \qquad \text{for } i \in \mathbb{Z}.$$

Note that there exist $l+1$ points in $U_\lambda(l)$ and hence

$$e(l) = \left\lfloor \frac{l+1}{2} \right\rfloor.$$

Let us consider the local configuration s_λ on the coordinate neighborhood $U_\lambda(l)$ of $x \in B$ in the cell $C(r)$

$$\begin{aligned} &s_\lambda(x + e(0)) = (x + e(0), f_{x+e(0)}), \\ &s_\lambda(x + e(1)) = (x + e(1), f_{x+e(1)}), \\ &\cdots \\ &s_\lambda(x + e(l)) = (x + e(l), f_{x+e(l)}), \end{aligned}$$

where $f_{x+e(0)}, f_{x+e(1)}, \cdots, f_{x+e(l)} \in \{0, 1, \ldots, r-1\}$. For simplicity, we often write this as follows

$$\begin{aligned} s_\lambda &= (f_{x+e(0)} f_{x+e(1)} \cdots f_{x+e(l)}) \\ &= \left[\sum_{j=0}^{l} r^j f_{x+e(l-j)} \right], \end{aligned}$$

where a number between round (resp. square) brackets is the r-nary (resp. decimal) expression. Then the local transition T_λ is given as follows

$$T_\lambda(s_\lambda)(x) = \delta(f_{x+e(0)}, f_{x+e(1)}, \cdots, f_{x+e(l)}) = \bar{f}_x \in F_x,$$

where $\delta : \{0, 1, \ldots, r-1\}^{l+1} \to \{0, 1, \ldots, r-1\}$ and $\bar{f}_x \in \{0, 1, \ldots, r-1\}$. We assume that the map δ does not depend on x, and hence each member of $\mathcal{W}(l, r)$ is homogeneous.

There exist r^{l+1} possibilities of choosing δ, hence we assign the number

$$\rho = \sum_{i_0, i_1, \cdots, i_l \in \{0,1,\ldots,r-1\}} \delta(i_0, i_1, \cdots, i_l) r^{\sum_{j=0}^{l} r^j i_{l-j}}$$

to the local transition given by δ; and the member of $\mathcal{W}(l, r)$ given by the local transition referred to the number ρ is called the rule ρ.

3.2. Correspondence to de Bruijn Quiver

Lemma 2. There exists a one-to-one correspondence between the local configuration $s_\lambda = (f_{x+e(0)} f_{x+e(1)} \cdots f_{x+e(l)})$ in $\mathcal{W}(l, r)$ and the arrow α_i in $\mathcal{B}(l, r)$ given by

$$i = \sum_{j=0}^{l} r^j f_{x+e(l-j)}.$$

(Proof) The number i stands for the decimal number whose r-nary expression is $f_{x+e(0)} f_{x+e(1)} \cdots f_{x+e(l)}$. Since i ranges over $\{0, 1, \ldots, r^{l+1} - 1\}$ without redundancy when $f_{x+e(0)}, f_{x+e(1)}, \cdots, f_{x+e(l)}$ range over $\{0, 1, \ldots, r-1\}$, the correspondence is one-to-one. □

Lemma 3. There exists a one-to-one correspondence between the local configuration $s_\lambda = (f_{x+e(0)} f_{x+e(1)} \cdots f_{x+e(l)})$ in $\mathcal{W}(l, r)$ and the pair (a_j, a_k) of vertices in $\mathcal{B}(l, r)$, where j and k are given as follows

$$j = \sum_{m=1}^{l} r^{m-1} f_{x+e(l-m)} \qquad \text{and} \qquad k = \sum_{m=0}^{l-1} r^m f_{x+e(l-m)}.$$

(Proof) By virtue of lemma 2, assume α_i to be the vertex corresponding to s_λ:

$$i = \sum_{m=0}^{l} r^m f_{x+e(l-m)}.$$

By definition of the de Bruijn quiver $\mathcal{B}(l, r)$, if we assume a_j to be the source of α_i then we have

$$j = \left\lfloor \frac{i}{r} \right\rfloor = \sum_{m=1}^{l} r^{m-1} f_{x+e(l-m)}.$$

On the other hand, if we assume a_k to be the target of α_i then we have

$$\begin{aligned} k &\equiv i \pmod{r^l} \\ &= \sum_{j=0}^{l-1} r^m f_{x+e(l-m)}. \end{aligned}$$

This completes the proof. □

Above lemmas 2 and 3 say that there exists a one-to-one correspondence between a configuration $s_\lambda = (f_{x+e(0)} f_{x+e(1)} \cdots f_{x+e(l)})$ in $\mathcal{W}(l, r)$ and a path $(a_{\lfloor i/r \rfloor} | \alpha_i | a_{\bar{i}})$ in $\mathcal{B}(l, r)$, where $\bar{i} \equiv i$ (mod r^l). We denote the correspondence by $H_\lambda : \Gamma_\lambda(E) \to K\mathcal{B}(l, r)_1$

$$s_\lambda = (f_{x+e(0)} f_{x+e(1)} \cdots f_{x+e(l)}) \mapsto (a_{\lfloor i/r \rfloor} | \alpha_i | a_{\bar{i}}), \tag{2}$$

where $i = \sum_{j=0}^{l} r^j f_{x+e(l-j)}$.

Let

$$\begin{aligned} U_{\underline{\lambda}}(l) &= \{x + e(0), x + e(1), \cdots, x + e(l-1)\} = U_\lambda(l) \backslash \{x + e(l)\}, \\ U_{\overline{\lambda}}(l) &= \{x + e(1), x + e(2), \cdots, x + e(l)\} = U_\lambda(l) \backslash \{x + e(0)\}. \end{aligned}$$

Also let the restriction s_λ to $U_{\underline{\lambda}}(l)$ and $U_{\overline{\lambda}}(l)$ be $s_{\underline{\lambda}}$ and $s_{\overline{\lambda}}$, respectively. These restrictions $s_{\underline{\lambda}}$ and $s_{\overline{\lambda}}$ of the local configuration s_λ are called configurations of length 0.

Example 10. Let us consider the WCA $\mathcal{W}(2, 2)$, which is nothing but the ECA. Then we have $E = \mathbb{Z} \times \{0, 1\}$, $B = \mathbb{Z}$, and $\sigma(x) = U_\lambda(2) = \{x - 1, x, x + 1\}$. All the possible local configurations are as follows

$$\begin{aligned} s_\lambda = (f_{x-1} f_x f_{x+1}) = {} & (000), (001), (010), (011), \\ & (100), (101), (110), (111). \end{aligned}$$

The correspondence between the local configurations in $\mathcal{W}(2, 2)$ and the arrows and the vertices in the de Bruijn quiver $\mathcal{B}(2, 2)$ can be summarized in the following table.

s_λ	(000)	(001)	(010)	(011)	(100)	(101)	(110)	(111)
$s_{\underline{\lambda}}$	(00)	(00)	(01)	(01)	(10)	(10)	(11)	(11)
$s_{\overline{\lambda}}$	(00)	(01)	(10)	(11)	(00)	(01)	(10)	(11)
arrow	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7
source	a_0	a_0	a_1	a_1	a_2	a_2	a_3	a_3
target	a_0	a_1	a_2	a_3	a_0	a_1	a_2	a_3

Let $\sigma(x + j) = U_{\lambda_j}(l)$ for $j = 0, 1, \ldots, n - 1$. Then we have

$$\begin{aligned} & s_{\lambda_j}(x + j + e(0)) = (x + j + e(0), f_{x+j+e(0)}) \\ & s_{\lambda_j}(x + j + e(1)) = (x + j + e(1), f_{x+j+e(1)}) \\ & \dots \\ & s_{\lambda_j}(x + j + e(l)) = (x + j + e(l), f_{x+j+e(l)}) \end{aligned}$$

for $j = 0, 1, \ldots, n-1$, and hence $s_{\lambda_j} = (f_{x+j+e(0)} f_{x+j+e(1)} \cdots f_{x+j+e(l)})$. For simplicity, we denote a restriction $s|_{U_{\lambda_0}(l)\cup U_{\lambda_1}(l)\cup\cdots\cup U_{\lambda_{n-1}}(l)}$ of a global configuration s in $\mathcal{W}(l, r)$ to an open set $U_{\lambda_0}(l) \cup U_{\lambda_1}(l) \cup \cdots \cup U_{\lambda_{n-1}}(l)$ by $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ and call it a configuration of length $n > 0$. We often write

$$\begin{aligned} s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} &= \prod_{j=0}^{n-1} (f_{x+j+e(0)} f_{x+j+e(1)} \cdots f_{x+j+e(l)}) \\ &= (f_{x+e(0)} f_{x+1+e(0)} \cdots f_{x+n-1+e(l)}) \\ &= \prod_{j=0}^{n-1} \left[\sum_{k=0}^{l} r^k f_{x+j+e(l-k)} \right]. \end{aligned}$$

We can extend $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ to $n = 0$ as if $n = 0$ then $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} = s_{\underline{\lambda_0}}$. Then, by virtue of lemma 3, there exists a one-to-one correspondence between $s_{\underline{\lambda}}$ and vertex a_j for $j = \sum_{m=1}^{l} r^{m-1} f_{e(l-m)}$ in $\mathcal{B}(l, r)$. Similarly, there also exists a one-to-one correspondence between $s_{\overline{\lambda}}$ and vertex a_k for $k = \sum_{m=0}^{l-1} r^m f_{e(l-m)}$ in $\mathcal{B}(l, r)$.

Remember that the global configuration s in $\mathcal{W}(l, r)$ is induced from the local configurations s_λ for $\lambda \in \Lambda$. Thus a one-to-one correspondence between the global configurations in $\mathcal{W}(l, r)$ and paths in $\mathcal{B}(l, r)$ is induced from the map H_λ for $\lambda \in \Lambda$.

Theorem 1. For any integer $n \geq 0$, there exists a one-to-one correspondence between a configuration $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ of length n in $\mathcal{W}(l, r)$ and a path $(a_{\lfloor i_0/r \rfloor} | \alpha_{i_0}\alpha_{i_1} \cdots \alpha_{i_{n-1}} | a_{\bar{i}_{n-1}})$ of length n in $\mathcal{B}(l, r)$, where $\bar{i}_n \equiv i_n$ (mod r^l) and $i_j = \sum_{k=0}^{l} r^k f_{x+j+e(l-k)}$.

(Proof) Note that $\sigma(x) = U_{\lambda_0}(l)$ is the coordinate neighborhood of $x \in B$ in $\mathcal{W}(l, r)$:

$$U_{\lambda_0}(l) = \{x + e(0), x + e(1), \cdots, x + e(l)\}.$$

Then, by virtue of lemma 2, there exists a unique arrow α_{i_0} corresponding to the local configuration s_{λ_0}, where i_0 is given as follows

$$i_0 = \sum_{m=0}^{l} r^m f_{x+e(l-m)}.$$

Let a_j be the target of α_{i_0}. Then, by virtue of lemma 3, j is uniquely determined as follows

$$j = \sum_{m=0}^{l-1} r^m f_{x+e(l-m)}.$$

On the other hand, j is also the source of an arrow α_{i_1} for

$$i_1 = \sum_{m=0}^{l-1} r^{m+1} f_{x+e(l-m)} + f_{x+e(l+1)} = \sum_{m=0}^{l} r^m f_{x+e(l-m+1)}.$$

Noting the following

$$\begin{aligned} r \cdot i_0 &= \sum_{m=0}^{l} r^{m+1} f_{x+e(l-m)} \\ &\equiv \sum_{m=1}^{l} r^{m} f_{x+e(l-m+1)} \pmod{r^{l+1}}, \end{aligned}$$

we have

$$i_1 \equiv r \cdot i_0 + f_{x+e(l+1)} \pmod{r^{l+1}}.$$

This implies that the arrow α_{i_0} is joined with the arrow α_{i_1} via the vertex a_j (see lemma 1). By virtue of lemma 2, there exists a unique local configuration s_{λ_1} on the coordinate neighborhood

$$\sigma(x+1) = U_{\lambda_1}(l) = \{x+1+e(0), x+1+e(1), \cdots, x+1+e(l)\}$$

of $x+1 \in B$; and s_{λ_1} corresponds to the vertex α_{i_1}. Thus the restriction $s_{\lambda_0\lambda_1}$ of the global transition s to $U_{\lambda_0}(l) \cup U_{\lambda_1}(l)$ corresponds to the path $\alpha_{i_0}\alpha_{i_1}$.

Thus we inductively obtain the one-to-one correspondence between a configuration $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ of length n in $\mathcal{W}(l,r)$ and a path $(a_{\lfloor i_0/r \rfloor}|\alpha_{i_0}\alpha_{i_0}\cdots\alpha_{i_{n-1}}|a_{\bar{i}_{n-1}})$ of length n in $\mathcal{B}(l,r)$. □

We denote the one-to-one correspondence given in theorem 1 by $H : \Gamma(E) \to K\mathcal{B}(l,r)$

$$H : s_{\lambda_0\cdots\lambda_{n-1}} = (f_{x+e(0)} \cdots f_{x+n-1+e(l)}) \mapsto (a_{\lfloor i_0/r \rfloor}|\alpha_{i_0}\cdots\alpha_{i_{n-1}}|a_{\bar{i}_{n-1}}), \tag{3}$$

where $i_j = \sum_{k=0}^{l} r^k f_{x+j+e(l-k)}$.

Example 11. Let us consider the WCA $\mathcal{W}(2,2)$ as in example 10. Assume $n = 3$ and consider configurations of length 3. Then the coordinate neighborhoods are

$$\begin{aligned} \sigma(x) &= U_{\lambda_0}(2) = \{x-1, x, x+1\}, \\ \sigma(x+1) &= U_{\lambda_1}(2) = \{x, x+1, x+2\}, \\ \sigma(x+2) &= U_{\lambda_2}(2) = \{x+1, x+2, x+3\}, \end{aligned}$$

and hence we have $U_{\lambda_0}(2) \cup U_{\lambda_1}(2) \cup U_{\lambda_2}(2) = \{x-1, x, x+1, x+2, x+3\}$. The configuration $s_{\lambda_0\lambda_1\lambda_2}$ of length 3 takes 32 binary numbers from (00000) to (11111).

The correspondence between the configurations of length 3 in $\mathcal{W}(2,2)$ and the paths of length 3 in $\mathcal{B}(2,2)$ is given in the following table.

(00000)	(00001)	(00010)	(00011)
$(a_0\|\alpha_0\alpha_0\alpha_0\|a_0)$	$(a_0\|\alpha_0\alpha_0\alpha_1\|a_1)$	$(a_0\|\alpha_0\alpha_1\alpha_2\|a_2)$	$(a_0\|\alpha_0\alpha_1\alpha_3\|a_3)$
(00100)	(00101)	(00110)	(00111)
$(a_0\|\alpha_1\alpha_2\alpha_4\|a_0)$	$(a_0\|\alpha_1\alpha_2\alpha_5\|a_1)$	$(a_0\|\alpha_1\alpha_3\alpha_6\|a_2)$	$(a_0\|\alpha_1\alpha_3\alpha_7\|a_3)$
(01000)	(01001)	(01010)	(01011)
$(a_1\|\alpha_2\alpha_4\alpha_0\|a_0)$	$(a_1\|\alpha_2\alpha_4\alpha_1\|a_1)$	$(a_1\|\alpha_2\alpha_5\alpha_2\|a_2)$	$(a_1\|\alpha_2\alpha_5\alpha_3\|a_3)$
(01100)	(01101)	(01110)	(01111)
$(a_1\|\alpha_3\alpha_6\alpha_4\|a_0)$	$(a_1\|\alpha_3\alpha_6\alpha_5\|a_1)$	$(a_1\|\alpha_3\alpha_7\alpha_6\|a_2)$	$(a_1\|\alpha_3\alpha_7\alpha_7\|a_3)$
(10000)	(10001)	(10010)	(10011)
$(a_2\|\alpha_4\alpha_0\alpha_0\|a_0)$	$(a_2\|\alpha_4\alpha_0\alpha_1\|a_1)$	$(a_2\|\alpha_4\alpha_1\alpha_2\|a_2)$	$(a_2\|\alpha_4\alpha_1\alpha_3\|a_3)$
(10100)	(10101)	(10110)	(10111)
$(a_2\|\alpha_5\alpha_2\alpha_4\|a_0)$	$(a_2\|\alpha_5\alpha_2\alpha_5\|a_1)$	$(a_2\|\alpha_5\alpha_3\alpha_6\|a_2)$	$(a_2\|\alpha_5\alpha_3\alpha_7\|a_3)$
(11000)	(11001)	(11010)	(11011)
$(a_3\|\alpha_6\alpha_4\alpha_0\|a_0)$	$(a_3\|\alpha_6\alpha_4\alpha_1\|a_1)$	$(a_3\|\alpha_6\alpha_5\alpha_2\|a_2)$	$(a_3\|\alpha_6\alpha_5\alpha_3\|a_3)$
(11100)	(11101)	(11110)	(11111)
$(a_3\|\alpha_7\alpha_6\alpha_4\|a_0)$	$(a_3\|\alpha_7\alpha_6\alpha_5\|a_1)$	$(a_3\|\alpha_7\alpha_7\alpha_6\|a_2)$	$(a_3\|\alpha_7\alpha_7\alpha_7\|a_3)$

A binary number stands for the configuration of length 3 which corresponds to the directly lower path in the above table.

Through the one-to-one correspondence given above, we can define the configuration algebra $K\mathcal{W}(l,r)$ of $\mathcal{W}(l,r)$ over an algebraically closed field K.

Definition 5. Let $\mathcal{W}(l,r)$ be the WCA. Also let K be an algebraically closed field. The configuration algebra $K\mathcal{W}(l,r)$ of $\mathcal{W}(l,r)$ is the K-algebra whose underlying K-vector space has as its basis the set of all configurations $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ of length $n \geq 0$ in $\mathcal{W}(l,r)$ and such that the product of two basis vectors $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ and $s_{\mu_0\mu_1\cdots\mu_{m-1}}$ of $K\mathcal{W}(l,r)$ is defined by

$$s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} s_{\mu_0\mu_1\cdots\mu_{m-1}} = \delta_{s_{\overline{\lambda_{n-1}}} s_{\underline{\mu_0}}} s_{\lambda_1\lambda_2\cdots\lambda_n\mu_0\mu_1\cdots\mu_{m-1}}.$$

The product of basis elements is then extended to arbitrary elements of $K\mathcal{W}(l,r)$ by distributivity.

The one-to-one correspondence (3) between a configuration $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ of length n in $\mathcal{W}(l,r)$ and a path $(a_{\lfloor i_0/r\rfloor}|\alpha_{i_0}\alpha_{i_1}\cdots\alpha_{i_{n-1}}|a_{\bar{i}_{n-1}})$ of length n in $\mathcal{B}(l,r)$ for $n \geq 0$ given in theorem 1 induces a bijection from the set of all configurations (*i.e.*, the basis of $K\mathcal{W}(l,r)$) to the set of all paths (*i.e.*, the basis of $K\mathcal{B}(l,r)$). We also denote the isomorphism from the configuration algebra of $\mathcal{W}(l,r)$ to the path algebra of $\mathcal{B}(l,r)$ induced from the bijection (3) by $H : K\mathcal{W}(l,r) \to K\mathcal{B}(l,r)$:

$$H(cs+dt) = c\alpha + d\beta, \qquad H(st) = \alpha\beta, \qquad H(0) = 0,$$

where s and t are configurations of length m and n in $K\mathcal{W}(l,r)$, respectively; and α and β are paths of length m and n in $K\mathcal{B}(l,r)$ which correspond to the configurations s and t in terms of (3), respectively.

Remark 1. The global transition $T:\Gamma(E)\to\Gamma(E)$ in the WCA $\mathcal{W}(l,r)$ does not concern the one-to-one correspondence $H: K\mathcal{W}(l,r)\to K\mathcal{B}(l,r)$.

3.3. Global Transition of Configuration Algebra

Now we consider the global transition $T:\Gamma(E)\to\Gamma(E)$ of the WCA $\mathcal{W}(l,r)$. We extend the global transition T to an endomorphism on the configuration algebra $K\mathcal{W}(l,r)$ of $\mathcal{W}(l,r)$ as follows. Denote the subspace of $K\mathcal{W}(l,r)$ generated by the set of all configurations of length $n\geq 0$ by $K\mathcal{W}(l,r)_n$. The global transition $T:\Gamma(E)\to\Gamma(E)$ of $\mathcal{W}(l,r)$ maps a configuration of length n into a configuration of the same length and the basis of $K\mathcal{W}(l,r)_n$ consists of all configurations of length n; therefore, T induces an endomorphism on $K\mathcal{W}(l,r)_n$ for any $n\geq 0$. Thus we obtain an endomorphism on the configuration algebra $K\mathcal{W}(l,r)$, which is also denoted by T:

$$T(cs+dt)=cT(s)+dT(t),\qquad T(st)=T(s)T(t),\qquad T(0)=0,$$

where $c,d\in K$ and s and t are configurations of length m and n in $K\mathcal{W}(l,r)$, respectively. We call $T\in\mathrm{End}(K\mathcal{W}(l,r))$ the global transition of the configuration algebra $K\mathcal{W}(l,r)$.

Then we can also define an endomorphism $\tilde{T}$ on the path algebra $K\mathcal{B}(l,r)$ of the de Bruijn quiver in order the following diagram to commute

$$\begin{tikzcd}
K\mathcal{W}(l,r) \arrow[r, "T"] \arrow[d, "H"'] & K\mathcal{W}(l,r) \arrow[d, "H"] \\
K\mathcal{B}(l,r) \arrow[r, "\tilde{T}"] & K\mathcal{B}(l,r).
\end{tikzcd}$$

We call $\tilde{T}\in\mathrm{End}(K\mathcal{B}(l,r))$ the global transition of the path algebra $K\mathcal{B}(l,r)$ associated with $\mathcal{W}(l,r)$.

Example 12. Let us consider the WCA $\mathcal{W}(2,2)$ as in example 11. Consider the local transition referred to $\rho=154$:

$$\begin{aligned}
&\delta(0,0,0)=0,\ \delta(0,0,1)=1,\ \delta(0,1,0)=0,\ \delta(0,1,1)=1,\\
&\delta(1,0,0)=1,\ \delta(1,0,1)=0,\ \delta(1,1,0)=0,\ \delta(1,1,1)=1.
\end{aligned}$$

Applying the global transition T induced from the above local transition to, say $(\cdots 00010111001\cdots)+(\cdots 00111101011\cdots)\in K\mathcal{W}(2,2)$, we obtain

$$\begin{aligned}
&(\cdots 00010111001\cdots)+(\cdots 00111101011\cdots)\\
\xrightarrow{T}\quad &(\cdots 110011011\cdots)+(\cdots 111100001\cdots).
\end{aligned}$$

This corresponds to the following transition of the path algebra $K\mathcal{B}(2,2)$

$$\begin{aligned}
&(\cdots\alpha_0\alpha_1\alpha_2\alpha_5\alpha_3\alpha_7\alpha_6\alpha_4\alpha_1\cdots)+(\cdots\alpha_1\alpha_3\alpha_7\alpha_7\alpha_6\alpha_5\alpha_2\alpha_5\alpha_3\cdots)\\
\xrightarrow{\tilde{T}}\quad &(\cdots\alpha_6\alpha_4\alpha_1\alpha_3\alpha_6\alpha_5\alpha_3\cdots)+(\cdots\alpha_7\alpha_7\alpha_6\alpha_4\alpha_0\alpha_0\alpha_1\cdots).
\end{aligned}$$

3.4. Transition Matrices

Let us define the adjacency matrix of the WCA via the isomorphism between the configuration algebra of the WCA and the path algebra of the de Bruijn quiver.

Definition 6. Let $\tilde{\mathcal{M}}$ be the adjacency matrix of the de Bruijn quiver $\mathcal{B}(l,r)$. Also let $\tilde{\mu}_{jk}$ be the (j,k)-entry of $\tilde{\mathcal{M}}$ for $j,k=1,2,\dots,r^l$. Then the adjacency matrix $\mathcal{M}=(\mu_{jk})$ of the WCA $\mathcal{W}(l,r)$ is defined as follows

$$\mu_{jk}=H^{-1}(\tilde{\mu}_{jk}) \qquad \text{for } j,k=1,2,\dots,r^l,$$

where $H: K\mathcal{W}(l,r)\to K\mathcal{B}(l,r)$ is the K-algebra isomorphism induced from the bijection (3) between the configurations in $\mathcal{W}(l,r)$ and the paths in $\mathcal{B}(l,r)$ of the same length.

Lemma 4. Let $\tilde{\mathcal{M}}=(\tilde{\mu}_{jk})$ and $\mathcal{M}=(\mu_{jk})$ be as in definition 6. Then we have

$$\mu_{jk}=\begin{cases}[i] & \text{if } \tilde{\mu}_{jk}=\alpha_i\\ 0 & \text{if } \tilde{\mu}_{jk}=0\end{cases} \qquad \text{for } j,k=1,2,\dots,r^l,$$

where $i=\sum_{j=0}^{l} r^j f_{x+e(l-j)}$, $s_\lambda=(f_{x+e(0)}f_{x+e(1)}\cdots f_{x+e(l)})$, and $H:(f_{x+e(0)}f_{x+e(1)}\cdots f_{x+e(l)})\mapsto(a_{\lfloor i/r\rfloor}|\alpha_i|a_{\bar{i}})$.

(Proof) Since $H:(f_{x+e(0)}f_{x+e(1)}\cdots f_{x+e(l)})\mapsto(a_{\lfloor i/r\rfloor}|\alpha_i|a_{\bar{i}})$, if $\tilde{\mu}_{jk}=\alpha_i$ then

$$\begin{aligned}\mu_{jk}&=H^{-1}(\tilde{\mu}_{jk})\\ &=(f_{x+e(0)}f_{x+e(1)}\cdots f_{x+e(l)})\\ &=\left[\sum_{j=0}^{l} r^j f_{x+e(l-j)}\right]=[i].\end{aligned}$$

Obviously, H^{-1} maps $0\in K\mathcal{B}(l,r)$ into $0\in K\mathcal{W}(l,r)$. □

We can obtain the explicit form of the trace of any power of the adjacency matrix of the WCA as a direct consequence of proposition 3 and lemma 4.

Proposition 4. Let $\mathcal{M}$ be the adjacency matrix of the WCA $\mathcal{W}(l,r)$. Then the trace of the n-th power of $\mathcal{M}$ is

$$\mathrm{tr}\,(\mathcal{M}^n)=\sum_{j_0,j_1,\cdots,j_{n-2}\in\{0,1,\dots,r^{l+1}-1\}}[j_0][j_1]\cdots[j_{n-1}],$$

where j_i ranges over $\{0,1,\dots,r^{l+1}-1\}$ with each successive pair (j_i,j_{i+1}) satisfying $j_{i+1}\equiv r\cdot j_i, r\cdot j_i+1,\cdots,r\cdot j_i+r-1$ (mod r^{l+1}) for $i=0,1,\dots,n-1$ and we assume $j_n=j_0$. □

We then define the transition matrix of the WCA by applying the global transition to the adjacency matrix of the WCA.

Definition 7. Let $\mathcal{M}=(\mu_{jk})$ be the adjacency matrix of the WCA $\mathcal{W}(l,r)$. Also let $T\in\mathrm{End}(K\mathcal{W}(l,r))$ be the global transition of $\mathcal{W}(l,r)$. We call $T\,(\mathcal{M}):=(T(\mu_{jk}))$ the transition matrix of $\mathcal{W}(l,r)$.

Example 13. Let A be the WCA $\mathcal{W}(l,2)$. Then the adjacency matrix $\mathcal{M}$ is the following $2^l \times 2^l$ matrix

$$\mathcal{M} = \begin{pmatrix} [0] & [1] & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & [2] & [3] & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & [2^l-2] & [2^l-1] \\ [2^l] & [2^l+1] & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & [2^l+2] & [2^l+3] & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cdots & [2^{l+1}-2] & [2^{l+1}-1] \end{pmatrix},$$

where we abbreviate the local configuration $s_\lambda = (f_{x+e(0)}f_{x+e(1)}\cdots f_{x+e(l)})$ to $\left[\sum_{j=0}^{l} r^j f_{x+e(l-j)}\right]$.

If we put $l=2$ we obtain

$$\mathcal{M} = \begin{pmatrix} [0] & [1] & 0 & 0 \\ 0 & 0 & [2] & [3] \\ [4] & [5] & 0 & 0 \\ 0 & 0 & [6] & [7] \end{pmatrix}.$$

Moreover, if we choose the local transition referred to $\rho = 154$ as in example 12:

$$\begin{aligned} &(000)=[0]\mapsto(0), \quad (001)=[1]\mapsto(1), \quad (010)=[2]\mapsto(0), \quad (011)=[3]\mapsto(1), \\ &(100)=[4]\mapsto(1), \quad (101)=[5]\mapsto(0), \quad (110)=[6]\mapsto(0), \quad (111)=[7]\mapsto(1), \end{aligned}$$

we obtain the following transition matrix $T(\mathcal{M})$ of $\mathcal{W}(2,2)$.

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (0) & (1) \\ (1) & (0) & 0 & 0 \\ 0 & 0 & (0) & (1) \end{pmatrix},$$

where the number (0) and (1) in the entries are $(\bar{f}_x) = (\delta(f_{x-1}, f_x, f_{x+1}))$ and $\delta : \{0,1\}^3 \to \{0,1\}$ is the local transition.

Remark 2. By definition of the adjacency matrix $\mathcal{M}$, every local configuration of length n obtained by applying the global transition T to some configuration in the WCA appears in $T(\mathcal{M})^n$ just once. Conversely, every terms in the entries of $T(\mathcal{M})^n$ is a local configuration of length n obtained by applying T to some configuration in the WCA.

Since the global transition T is a K-algebra endomorphism on $K\mathcal{W}(l,r)$, we have

$$T\left(\operatorname{tr}\left(\mathcal{M}^n\right)\right) = \operatorname{tr}\left(T(\mathcal{M})^n\right).$$

Thus the global transition of the trace of the n-th power of the adjacency matrix $\mathcal{M}$ is the trace of the n-th power of the transition matrix $T(\mathcal{M})$.

4. Reversibility of Cellular Automata

Now let us consider reversibility of cellular automata. If the global transition $T : \Gamma(E) \to \Gamma(E)$ of a cellular automaton $A = (C, \{U_\lambda\}_{\lambda\in\Lambda}, \sigma, T)$ is bijective then the cellular automaton A is said to be reversible. Hereafter, we study reversibility of the WCA $\mathcal{W}(l,r)$ via the correspondence to the de Bruijn quiver $\mathcal{B}(l,r)$.

Under the above definition of reversibility of cellular automata, however, almost all ones are not reversible. This is because the WCA $\mathcal{W}(l,r)$ is so large as to contain all possible configurations of arbitrary length. Therefore, we may reduce $\mathcal{W}(l,r)$ to a certain subclass of it to detect the behavior of each cellular automaton with respect to the global transition imposing appropriate boundary conditions. Noticing that the global transition T of $\mathcal{W}(l,r)$ is an endomorphism on the configuration algebra $K\mathcal{W}(l,r) = \bigoplus_{n=0}^{\infty} K\mathcal{W}(l,r)_n$, we have

$$T\left(\bigoplus_{n=0}^{\infty} K\mathcal{W}(l,r)_n\right) = \bigoplus_{n=0}^{\infty} T\left(K\mathcal{W}(l,r)_n\right),$$

where $K\mathcal{W}(l,r)_n$ is the sub-vector space of $K\mathcal{W}(l,r)$ consisting of configurations of length n. Thus it is natural to consider reversibility of a subclass of $\mathcal{W}(l,r)$ consisting of configurations of a certain length.

In order to detect the behavior of a subclass of the WCA $\mathcal{W}(l,r)$ consisting of configurations of length n with respect to the global transition, we need to impose appropriate boundary conditions. Although there can be many kinds of boundary conditions, the most natural one may be periodic boundary conditions. Therefore, hereafter, we consider the WCA consisting of configurations of a certain length imposing the periodic boundary conditions.

4.1. Periodic Reductions of WCA

Let $\mathcal{W}(l,r) = (C(r), \{U_\lambda(l)\}_{\lambda\in\Lambda}, \sigma, T)$ be the WCA. Let n be a natural number not less than l. Let B_n be the subspace $\mathbb{Z}/n\mathbb{Z} = \{0, , 1, \ldots, n-1\}$ of the base space $B = \mathbb{Z}$ of the cell $C(r)$. Assume the coordinate neighborhood $\sigma(x)$ of $x \in B_n$ is given as follows

$$\sigma(x) = U_\lambda(l) = \{\bar{e}(x), \bar{e}(x+1), \cdots, \bar{e}(x+l)\}, \tag{4}$$

where $\bar{e} : \mathbb{Z} \to \mathbb{Z}$ is defined by

$$\bar{e}(i) \equiv -\left\lfloor \frac{l}{2} \right\rfloor + i \pmod{n} \qquad \text{for } i \in \mathbb{Z}.$$

We call such $\mathcal{W}(l,r)$ that the base space is restricted to B_n with the choice (4) of σ the n-periodic reduction of the WCA, and abbreviate it to the n-WCA and denote it by $\mathcal{W}(l,r)_n$.

Let $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ be a configuration of length n in the n-WCA. Then we have

$$\begin{aligned}
&\sigma(0) = U_{\lambda_0}(l) = \{\bar{e}(0), \bar{e}(1), \cdots, \bar{e}(l)\},\\
&\sigma(1) = U_{\lambda_1}(l) = \{\bar{e}(1), \bar{e}(2), \cdots, \bar{e}(1+l)\},\\
&\qquad\vdots\\
&\sigma(n-l-1) = U_{\lambda_{n-l-1}}(l) = \{\bar{e}(n-l-1), \bar{e}(n-l), \cdots, \bar{e}(n-1)\},\\
&\qquad\vdots\\
&\sigma(n-1) = U_{\lambda_{n-1}}(l) = \{\bar{e}(n-1), \bar{e}(n), \cdots, \bar{e}(n-1+l)\}
\end{aligned}$$

and $s_{\lambda_j} = (f_{\bar{e}(j)} f_{\bar{e}(j+1)} \cdots f_{\bar{e}(j+l)})$, *i.e.*,

$$\begin{aligned}
&s_{\lambda_j}(\bar{e}(j)) = (\bar{e}(j), f_{\bar{e}(j)})\\
&s_{\lambda_j}(\bar{e}(j+1)) = (\bar{e}(j+1), f_{\bar{e}(j+1)})\\
&\cdots\\
&s_{\lambda_j}(\bar{e}(j+l)) = (\bar{e}(j+l), f_{\bar{e}(j+l)}),
\end{aligned}$$

where $f_{\bar{e}(j)}, f_{\bar{e}(j+1)}, \cdots, f_{\bar{e}(j+l)} \in \{0, 1, \ldots, r-1\}$ for $j = 0, 1, \ldots, n-1$. Then we have

$$\begin{aligned}
s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} &= \prod_{j=0}^{n-1} (f_{\bar{e}(j)} f_{\bar{e}(j+1)} \cdots f_{\bar{e}(j+l)})\\
&= (f_{\bar{e}(0)} f_{\bar{e}(1)} \cdots f_{\bar{e}(n-1)} \cdots f_{\bar{e}(n-1+l)}).
\end{aligned}$$

We have the following lemma.

Lemma 5. A configuration $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ of lenth n in the n-WCA $\mathcal{W}(l, r)_n$ is isomorphic to the configuration $s_{\lambda_0\lambda_1\cdots\lambda_{n-l-1}}$ of length $n-l$ in the WCA $\mathcal{W}(l, r)$:

$$s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} \simeq s_{\lambda_0\lambda_1\cdots\lambda_{n-l-1}}.$$

(Proof) By definition of $\bar{e} : \mathbb{Z} \to \mathbb{Z}$, we have

$$f_{\bar{e}(n-1+i)} = f_{\bar{e}(i-1)} \qquad \text{for } i = 1, 2, \ldots, l.$$

Therefore there exists an identification $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} = (f_{\bar{e}(0)} f_{\bar{e}(1)} \cdots f_{\bar{e}(n-1+l)}) \simeq (f_{\bar{e}(0)} f_{\bar{e}(1)} \cdots f_{\bar{e}(n-1)}) = s_{\lambda_0\lambda_1\cdots\lambda_{n-l-1}}$. □

Let the subspace of the configuration algebra $K\mathcal{W}(l, r)$ spanned by the configurations in n-WCA be $K\mathcal{W}(l, r)'_n$. It directly follows from lemma 5 that the following proposition holds.

Proposition 5. The subspace $K\mathcal{W}(l, r)'_n$ of $K\mathcal{W}(l, r)$ consisting of the configurations of length n in the n-WCA $\mathcal{W}(l, r)$ is isomorphic to the subspace $K\mathcal{W}(l, r)_{n-l}$ of $K\mathcal{W}(l, r)$ consisting of the configurations of length $n-l$ in the WCA $\mathcal{W}(l, r)$:

$$K\mathcal{W}(l, r)'_n \simeq K\mathcal{W}(l, r)_{n-l}.$$

(Proof) The basis of $K\mathcal{W}(l,r)'_n$ consists of r^n configurations of length n in the n-WCA $\mathcal{W}(l,r)_n$. On the other hand, the basis of $K\mathcal{W}(l,r)_{n-l}$ consists of $r^{(n-l)+l} = r^n$ configurations of length $n-l$ in the WCA $\mathcal{W}(l,r)$ not reduced to $\mathcal{W}(l,r)_n$ (see corollary 1, 2 and theorem 1). By virtue of lemma 5, there exists a bijection between the above bases

$$s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} = \left(f_{\bar{e}(0)}f_{\bar{e}(1)}\cdots f_{\bar{e}(n-1+l)}\right) \mapsto \left(f_{\bar{e}(0)}\cdots f_{\bar{e}(n-1)}\right) = s_{\lambda_0\lambda_1\cdots\lambda_{n-l-1}},$$

which can be extended to the desired isomorphism. □

Note we have

$$s_{\overline{\lambda_0}} = \left(f_{\bar{e}(0)}\cdots f_{\bar{e}(l-1)}\right) = s_{\underline{\lambda_{n-1}}}.$$

Then, by theorem 1, the following proposition holds.

Proposition 6. A configuration $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}}$ of length n in the n-WCA is mapped into a cycle of length n in the de Bruijn quiver $\mathcal{B}(l,r)$ in terms of the K-algebra isomorphism $H : K\mathcal{W}(l,r) \to K\mathcal{B}(l,r)$

$$s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} = \left(f_{\bar{e}(0)}f_{\bar{e}(1)}\cdots f_{\bar{e}(n-1)}\right) \mapsto \left(a_{\lfloor i_0/r\rfloor}|\alpha_{i_0}\alpha_{i_1}\cdots\alpha_{i_{n-1}}|a_{\lfloor i_0/r\rfloor}\right),$$

where $i_j = \sum_{k=0}^{l} r^k f_{\bar{e}(j+l-k)}$ for $j = 0,1,\ldots,n-1$. □

Remark 3. Since we assume $n \geq l$, by virtue of corollary 1, there exist r^n cycles of length n in $\mathcal{B}(l,r)$. On the other hand, we have $s_{\lambda_0\lambda_1\cdots\lambda_{n-1}} = \left(f_{\bar{e}(0)}f_{\bar{e}(1)}\cdots f_{\bar{e}(n-1)}\right)$, and each f ranges over $\{0,1,\ldots,r-1\}$. Therefore, there exist r^n configurations in the n-WCA.

Example 14. Let us consider the 3-WCA $\mathcal{W}(2,2)_3$ (*cf.* example 11). Then the coordinate neighborhoods are

$$\begin{aligned}\sigma(0) &= U_{\lambda_0}(2) = \{2,0,1\} = B_3\\ \sigma(1) &= U_{\lambda_1}(2) = \{0,1,2\} = B_3\\ \sigma(2) &= U_{\lambda_2}(2) = \{1,2,0\} = B_3.\end{aligned}$$

Thus the coordinate neighborhoods are the restricted base space B_3 itself, and the configuration of length 3 is $s_{\lambda_0\lambda_1\lambda_2} = (f_0f_1f_2f_0f_1) \simeq (f_0f_1f_2) = s_{\lambda_0}$.

The correspondence between 2^3 configurations in $\mathcal{W}(2,2)_3$ and 2^3 cycles of length 3 in $\mathcal{B}(2,2)$ is given in the following table.

$(00000) \simeq (000)$	$(00100) \simeq (001)$	$(01001) \simeq (010)$	$(01101) \simeq (011)$
$(a_0\|\alpha_0\alpha_0\alpha_0\|a_0)$	$(a_0\|\alpha_1\alpha_2\alpha_4\|a_0)$	$(a_1\|\alpha_2\alpha_4\alpha_1\|a_1)$	$(a_1\|\alpha_3\alpha_6\alpha_5\|a_1)$
$(10010) \simeq (100)$	$(10110) \simeq (101)$	$(11011) \simeq (110)$	$(11111) \simeq (111)$
$(a_2\|\alpha_4\alpha_1\alpha_2\|a_2)$	$(a_2\|\alpha_5\alpha_3\alpha_6\|a_2)$	$(a_3\|\alpha_6\alpha_5\alpha_3\|a_3)$	$(a_3\|\alpha_7\alpha_7\alpha_7\|a_3)$

All configurations in $\mathcal{W}(2,2)_3$ appear in the trace $\mathrm{tr}\left(\mathcal{M}^3\right)$ of the third power of the adjacency matrix $\mathcal{M}$ of $\mathcal{W}(2,2)_3$:

$$\begin{aligned}\mathrm{tr}\left(\mathcal{M}^3\right) &= [0][0][0] + [1][2][4] + [2][4][1] + [3][6][5]\\ &\qquad + [4][1][2] + [5][3][6] + [6][5][3] + [7][7][7]\\ &= (000) + (001) + (011) + (100) + (110) + (010) + (001) + (111).\end{aligned}$$

Note that the configurations in the first and second lines of the above $\mathrm{tr}\left(\mathcal{M}^3\right)$ are the configurations of length 3 in $\mathcal{W}(2,2)$, which is not reduced to $\mathcal{W}(2,2)_3$ (see proposition 5).

4.2. Reversibility of n-WCA

Let us consider the global transition $T : K\mathcal{W}(l,r) \to K\mathcal{W}(l,r)$ of the configuration algebra $K\mathcal{W}(l,r)$. Also denote the restriction of T on the subspace $K\mathcal{W}(l,r)'_n$ of $K\mathcal{W}(l,r)$ by the same T; and call it the global transition of $K\mathcal{W}(l,r)'_n$. By definition, the restricted K-algebra endomorphism T satisfies $T\left(K\mathcal{W}(l,r)'_n\right) \subset K\mathcal{W}(l,r)'_n$. Therefore, we define reversibility of the n-WCA $\mathcal{W}(l,r)_n$ as follows.

Definition 8. The n-WCA $\mathcal{W}(l,r)_n$ is reversible if and only if the global transition $T \in \mathrm{End}\left(K\mathcal{W}(l,r)'_n\right)$ is surjective.

Let $\mathcal{M}$ be the adjacency matrix of the n-WCA $\mathcal{W}(l,r)_n$. Note the trace $\mathrm{tr}(\mathcal{M}^n)$ of the n-th power of $\mathcal{M}$ to be the sum of all configurations in $\mathcal{W}(l,r)_n$. Then we obtain the following theorem concerning reversibility of the n-WCA.

Theorem 2. Let $\mathcal{M}$ be the adjacency matrix of the n-WCA $\mathcal{W}(l,r)_n$. Then $\mathcal{W}(l,r)_n$ is reversible if and only if the following holds

$$T\left(\mathrm{tr}(\mathcal{M}^n)\right) = \mathrm{tr}\left(T(\mathcal{M})^n\right) = \mathrm{tr}(\mathcal{M}^n), \tag{5}$$

where $\mathrm{tr}(\mathcal{M}^n)$ is the trace of the n-th power of $\mathcal{M}$ and $T(\mathcal{M})$ is the transition matrix.

(Proof) The trace $\mathrm{tr}(\mathcal{M}^n)$ of the n-th power of $\mathcal{M}$ consists of the sum of all configurations in the n-WCA $\mathcal{W}(l,r)_n$. Therefore, if T surjective then (5) holds. Conversely, if (5) holds then $T\left(\mathrm{tr}(\mathcal{M}^n)\right)$ consists of the sum of all configurations in $\mathcal{W}(l,r)_n$. Therefore T is surjective. □

Example 15. Let us consider the 3-WCA $\mathcal{W}(2,2)_3$ as in example 14. Choose the local transition referred to $\rho = 154$ as in example 13:

$$\begin{array}{llll} (000)=[0]\mapsto(0), & (001)=[1]\mapsto(1), & (010)=[2]\mapsto(0), & (011)=[3]\mapsto(1), \\ (100)=[4]\mapsto(1), & (101)=[5]\mapsto(0), & (110)=[6]\mapsto(0), & (111)=[7]\mapsto(1). \end{array}$$

Then, since $T \in \mathrm{End}\left(K\mathcal{W}(l,r)\right)$, we obtain

$$\begin{aligned} T\left(\mathrm{tr}(\mathcal{M}^3)\right) &= \mathrm{tr}\left(T(\mathcal{M})^3\right) \\ &= (000)+(001)+(011)+(100)+(110)+(010)+(001)+(111) \\ &= \mathrm{tr}(\mathcal{M}^3). \end{aligned}$$

Thus the transition T conserves $\mathrm{tr}(\mathcal{M}^3)$ and hence it is reversible.

On the contrary, the 4-WCA $\mathcal{W}(2,2)_4$ with the same local transition is not reversible. Indeed, we can calculate $T\left(\mathrm{tr}(\mathcal{M}^4)\right)$ as follows

$$\begin{aligned} T\left(\mathrm{tr}(\mathcal{M}^4)\right) =& 3(0000)+(0011)+2(0101)+(0110)+(0111)+(1001) \\ &+2(1010)+(1011)+(1100)+(1101)+(1110)+(1111), \end{aligned}$$

and hence $T\left(\mathrm{tr}(\mathcal{M}^4)\right) \neq \mathrm{tr}(\mathcal{M}^4) = \sum_{i=0}^{2^4-1}[i]$.

4.3. Necessary Conditions for Reversibility of n-WCA

Now we give two necessary conditions for reversibility of a rule in the n-WCA. At first, we show two lemmas concerning the number of configurations denoted by an r-nary number $(i) \in \{(0), (1), \cdots, (r-1)\}$ in the transition matrix $T(\mathcal{M})$ to give the first necessary condition for reversibility of the n-WCA.

Lemma 6. Let $\mathcal{M}$ be the adjacency matrix of the n-WCA $\mathcal{W}(l,r)_n$. Let $T(\mathcal{M})$ be the transition matrix of $\mathcal{W}(l,r)_n$. Let $N_i(m)$ be the number of configurations of length 1 denoted by an r-nary number (i) in $T(\mathcal{M})^m$ for $m \geq 1$ and $i = 0, 1, \ldots, r-1$. Then we have

$$N_i(m) = mr^{m-1}N_i(1) \qquad \text{for } i = 0, 1, \ldots, r-1 \text{ and } m \geq l.$$

(Proof) Since $m \geq l$, every entry in $T(\mathcal{M})^m$ is not 0. Let the (i,j)-entry in $T(\mathcal{M})$ be $b_{ij} \in \{0, (0), (1), \cdots, (r-1)\}$. Note that there exist r nonzero elements in each row of $T(\mathcal{M})$. Let μ_{ij} be the (i,j)-entry of $T(\mathcal{M})^m$ for $m \geq l$. Then we have $T(\mathcal{M})^{m+1} = T(\mathcal{M})^m T(\mathcal{M}) = \left(\sum_{k=1}^{r^l} \mu_{ik} b_{kj}\right)$. Noting that there exist r^{m-l} terms in every μ_{ij} (see proposition 2), we obtain the following recurrence equation for $N_i(m)$

$$\begin{aligned} N_i(m+1) &= rN_i(m) + r^l N_i(1) \times r^{m-l} \\ &= rN_i(m) + r^m N_i(1). \end{aligned}$$

Because there exist r nonzero b_{kj} in every column of $T(\mathcal{M})$ and there exist r^{m-l} terms in every μ_{ij}, the (i)'s in $N_i(m+1)$ from $T(\mathcal{M})$ is $rN_i(m)$ and from $T(\mathcal{M})^m$ is $r^l N_i(1) \times r^{m-l}$. Solving the above recurrence equation completes the proof. □

Lemma 7. Let $\mathcal{M}$ be the adjacency matrix of the n-WCA $\mathcal{W}(l,r)_n$. Let $T(\mathcal{M})$ be the transition matrix of $\mathcal{W}(l,r)_n$. Let $L_i(m)$ be the number of configurations of length 1 denoted by an r-nary number (i) in $\mathrm{tr}\,(T(\mathcal{M})^m)$ for $m \geq 1$ and $i = 0, 1, \ldots, r-1$. Then we have

$$L_i(m) = mr^{m-l-1}N_i(1) \qquad \text{for } i = 0, 1, \ldots, r-1 \text{ and } m \geq l,$$

where $N_i(1)$ is the number of (i)'s in $T(\mathcal{M})$ for $i = 0, 1, \ldots, r-1$.

(Proof) Let μ_{ij} be the same as in the proof of lemma 6. Also let $c_{ij} \in \{(0), (1), \cdots, (r-1)\}$ be the (i,j)-entry in $T(\mathcal{M})^l$. Note all entries c_{ij} in $T(\mathcal{M})^l$ to be nonzero. Then we have $\mathrm{tr}\left(T(\mathcal{M})^{m+l}\right) = \mathrm{tr}\left(T(\mathcal{M})^m T(\mathcal{M})^l\right) = \sum_{i=1}^{r^l} \sum_{k=1}^{r^l} \mu_{ik} c_{ki}$. Since there exist r^{2l} configurations of length l in $T(\mathcal{M})^l$ and there exist r^{l+1} nonzero entries in $T(\mathcal{M})$, the number of each (i) contained in $\mathrm{tr}\left(T(\mathcal{M})^{m+l}\right)$ is

$$l \times \frac{r^{2l}}{r^{l+1}} = lr^{l-1}.$$

Therefore we obtain

$$\begin{aligned} L_i(m+l) &= N_i(m) + lr^{l-1}N_i(1) \times r^{m-l} \\ &= mr^{m-1}N_i(1) + lr^{m-1}N_i(1) \\ &= (m+l)r^{m-1}N_i(1), \end{aligned}$$

where we use lemma 6. □

Then we obtain the first necessary condition for reversibility of a rule in the n-WCA $\mathcal{W}(l,r)_n$ concerning the number of an r-nary number $(i) \in \{(0),(1),\cdots,(r-1)\}$ in the transition matrix $T(\mathcal{M})$.

Theorem 3. Let $\mathcal{M}$ be the adjacency matrix of the n-WCA $\mathcal{W}(l,r)_n$ with the local transition referred to the rule ρ. Also let $T\left(\mathcal{M}\right)$ be the transition matrix of $\mathcal{W}(l,r)_n$. If the rule referred toρ is reversible then the transition matrix $T(\mathcal{M})$ contains exactly r^l configurations of length 1 denoted by an r-nary number (i) for $i = 0, 1, \ldots, r-1$.

(Proof) The $m(\geq l)$-th power $\mathcal{M}^m$ of the adjacency matrix $\mathcal{M}$ contains all r^m configurations in $\mathcal{W}(l,r)_n$ in its diagonal part. Therefore the number of (i)'s in $\mathrm{tr}\left(\mathcal{M}^m\right)$ is $m \times r^{m-1}$. On the other hand, by lemma 7, the number of (i)'s in $\mathrm{tr}\left(T(\mathcal{M})^m\right)$ is $nr^{m-l-1}N_i(1)$, where $N_i(1)$ is the number of (i)'s in $T(\mathcal{M})$ for $i = 0, 1, \cdots, r-1$. By virtue of theorem 2, if the rule referred to ρ is reversible then we have

$$nr^{n-1} = nr^{n-l-1}N_i(1).$$

Therefore we obtain $N_i(1) = r^l$. □

We also obtain another necessary condition for reversibility of a rule in the n-WCA $\mathcal{W}(l,r)_n$, which is useful to examine non-reversibility of a given rule in $\mathcal{W}(l,r)_n$.

Theorem 4. Let $\mathcal{M}$ be the adjacency matrix of the n-WCA $\mathcal{W}(l,r)_n$ with the local transition referred to the rule ρ. Also let $T\left(\mathcal{M}\right)$ be the transition matrix of $\mathcal{W}(l,r)_n$. If there exists a natural number m such that the m-th power $T\left(\mathcal{M}\right)^m$ of the transition matrix contains a configuration of length m plurally then the rule referred to ρ is not reversible except for finite n.

(Proof) Without loss of generality, we can assume the $(1,j)$-entry of $T\left(\mathcal{M}\right)^m$ is of the form

$$\kappa(\nu_1\nu_2\cdots\nu_m) + \cdots ,$$

where κ is a natural number greater than 1 and $\nu_1, \nu_2, \cdots, \nu_m \in \{(0),(1),\cdots,(r-1)\}$. Because if the (i,j)-entry of $T\left(\mathcal{M}\right)^m$ has the above form then the $(1,j)$ entry of $T\left(\mathcal{M}\right)^{m+l}$ has the form

$$\kappa(\xi_1\xi_2\cdots\xi_l\nu_1\nu_2\cdots\nu_m) + \cdots ,$$

where $(\xi_1\xi_2\cdots\xi_l)$ is the $(1,i)$-entry of $T\left(\mathcal{M}\right)^l$ and $\xi_1, \xi_2, \cdots, \xi_l \in \{(0),(1),\cdots,(r-1)\}$. (Note every entry of $T\left(\mathcal{M}\right)^l$ to be nonzero.)

Now we assume the $(j,1)$-entry of $T\left(\mathcal{M}\right)^l$ to be $(\zeta_1\zeta_2\cdots\zeta_l)$, where $\zeta_1, \zeta_2, \cdots, \zeta_l \in \{(0),(1),\cdots,(r-1)\}$. Then the $(1,1)$-entry of $T\left(\mathcal{M}\right)^{m+l}$ has the form

$$\kappa(\nu_1\nu_2\cdots\nu_m\zeta_1\zeta_2\cdots\zeta_l) + \cdots .$$

Hence the $(1,1)$-entry of $T\left(\mathcal{M}\right)^{m+l+k}$ has the form

$$\kappa\gamma(\nu_1\nu_2\cdots\nu_m\zeta_1\zeta_2\cdots\zeta_l\cdots) + \cdots$$

for $k \geq 1$, where $\gamma \geq 1$ is a natural number, because the $(1,1)$-entry of $T\left(\mathcal{M}\right)^k$ is never 0. Thus the trace of $T\left(\mathcal{M}\right)^{m+l+k}$ contains a configuration plurally for any $k \geq 0$, therefore the rule referred to ρ is not reversible for $n \geq m+l$. □

From example 15 and theorem 4, we see that reversibility of the n-WCA $\mathcal{W}(l,r)_n$ depends not only on its structure as a cellular automaton $\mathcal{W}(l,r) = (C(r), \{U_\lambda(l)\}_{\lambda\in\Lambda}, \sigma, T)$ but also on its period n. Thus it is important to study whether $\mathcal{W}(l,r)_n$ is reversible for a given natural number n not less than l. In the next section, we show that the n-WCA $\mathcal{W}(2,2)_n$ imposing periodic boundary conditions has 16 choices of the local transitions which are reversible for infinitely many natural numbers n.

5. Reversible Rules in ECA

In this subsection, we concentrate on the study of reversibility of the n-WCA $\mathcal{W}(2,2)_n$, which is nothing but the ECA imposing periodic boundary conditions firstly introduced by Wolfram [42]. Let the local transition T_λ of a local configuration $s_\lambda = (f_{\bar{e}(x)}f_{\bar{e}(x+1)}f_{\bar{e}(x+2)})$ on the coordinate neighborhood $\sigma(x) = U_\lambda = \{\bar{e}(x), \bar{e}(x+1), \bar{e}(x+2)\}$ be

$$\delta(f_{\bar{e}(x)}, f_{\bar{e}(x+1)}, f_{\bar{e}(x+2)}) = \bar{f}_x,$$

where $\delta : \{0,1\}^3 \to \{0,1\}$. Since we assume the local transition $\delta : \{0,1\}^3 \to \{0,1\}$ to be homogeneous, there exist 256 choices of δ in $\mathcal{W}(2,2)_n$, each of which is referred the number

$$\rho = \sum_{i_0,i_1,i_2\in\{0,1\}} \delta(i_0,i_1,i_2) 2^{2^2 i_0 + 2^1 i_1 + 2^0 i_2},$$

and called the rule ρ.

The global transition T of $\mathcal{W}(2,2)_n$ is induced from the local transition, and can be expressed by using the transition matrix $T(\mathcal{M})$

$$T(\mathcal{M}) = \begin{pmatrix} (\delta(0,0,0)) & (\delta(0,0,1)) & 0 & 0 \\ 0 & 0 & (\delta(0,1,0)) & (\delta(0,1,1)) \\ (\delta(1,0,0)) & (\delta(1,0,1)) & 0 & 0 \\ 0 & 0 & (\delta(1,1,0)) & (\delta(1,1,1)) \end{pmatrix}.$$

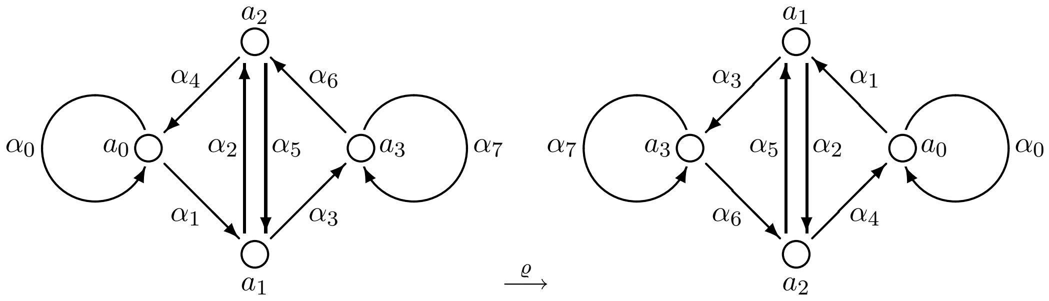
5.1. Equivalence Classes of Rules

By virtue of theorem 3, one of the necessary condition for reversibility of $\mathcal{W}(2,2)_n$ is that “the number of (0)’s in $T(\mathcal{M})$ is exactly 4”. Therefore, there exist ${}_8C_4 = 70$ choices of the rules ρ satisfying the necessary condition. Thus we may search 70 rules for reversible ones. Since the number 70 of rules is still so large, we introduce an equivalence relation among the rules and consider their equivalence classes.

Let us consider the graph automorphism $\varrho : \mathcal{B}(2,2) \to \mathcal{B}(2,2)$ of the de Bruijn quiver $\mathcal{B}(2,2) = (Q_0 = (a_0, \cdots, a_3), Q_1 = (\alpha_0, \cdots, \alpha_7), s, t)$ as follows

$$\varrho(a_i, \alpha_j) = (a_{3-i}, \alpha_{7-j}) \qquad \text{for } i = 0,1,2,3 \text{ and } j = 0,1,\ldots,7,$$

where the subscripts of a and α are considered modulo 4 and 8, respectively.



The graph automorphism ϱ induces an automorphism on the set of the local configurations s_λ in the n-WCA $\mathcal{W}(2,2)_n$ which is also denoted by ϱ

$$\varrho : s_\lambda = (f_0 f_1 f_2) \mapsto (\bar{f}_0 \bar{f}_1 \bar{f}_2),$$

where $\bar{f}_i = 1 - f_i$ for $i = 0, 1, 2$. The action of ϱ on the local transition T_λ in $\mathcal{W}(2,2)_n$ can be defined by

$$\left(\varrho \cdot T_\lambda\right)(s_\lambda) = T_\lambda\left(\varrho(s_\lambda)\right).$$

Then ϱ acts on ρ as follows

$$\begin{aligned}\varrho \cdot \rho &= \sum_{i_0,i_1,i_2\in\{0,1\}} \delta(1-i_0, 1-i_1, 1-i_2) 2^{2^{2}i_0+2^{1}i_1+2^{0}i_2} \\ &= \sum_{i_0,i_1,i_2\in\{0,1\}} \delta(i_0, i_1, i_2) 2^{7-(2^{2}i_0+2^{1}i_1+2^{0}i_2)}\end{aligned}$$

There exist 16 rules invariant under the action of the graph automorphism ϱ; and 6 of them, $\rho = 60, 90, 102, 153, 165, 195$, satisfy the necessary condition in theorem 3 for reversibility. Remaining $70 - 6 = 64$ rules can be identified with each other under the action of ϱ. Thus we may consider the following $32 + 6 = 38$ rules

$240,\ 232,\ 228,\ 226,\ 225,\ 216,\ 212,\ 210,\ 209,\ 204,\ 202,\ 201,\ 198,\ 197,\ 195,\ 184,$
$180,\ 178,\ 177,\ 172,\ 170,\ 169,\ 166,\ 165,\ 156,\ 154,\ 153,\ 150,\ 142,\ 120,\ 116,\ 114,$
$108,\ 106,\ 102,\ 92,\ 90,\ 60.$

Let us introduce two more automorphisms acting on the set of the rules ρ in $\mathcal{W}(2,2)_n$. One is the reflection ω which acts on the local configurations as follows

$$\omega : s_\lambda = (f_0 f_1 f_2) \mapsto (f_2 f_1 f_1).$$

Defining the action of ω on T_λ

$$\left(\omega \cdot T_\lambda\right)(s_\lambda) = T_\lambda\left(\omega(s_\lambda)\right),$$

it follows the action of ω on ρ

$$\omega \cdot \rho = \sum_{i_0,i_1,i_2\in\{0,1\}} \delta(i_2, i_1, i_0) 2^{2^{2}i_0+2^{1}i_1+2^{0}i_2}.$$

There exist 8 rules, $\rho = 232, 204, 201, 178, 165, 150, 108, 90$, invariant under the action of ω among the above 38 rules. There also exist 2 rules, $\rho = 209, 116$, invariant under the action of $\varrho \circ \omega$ among them. Remaining $38 - 10 = 28$ rules can be identified with each other under the action of ω. Thus we may consider the following $14 + 10 = 24$ rules.

$$\begin{gathered}240,\ 232,\ 228,\ 226,\ 225,\ 216,\ 212,\ 209,\ 204,\ 201,\ 198,\ 197,\\ 195,\ 180,\ 178,\ 165,\ 154,\ 150,\ 120,\ 116,\ 114,\ 108,\ 102,\ 90.\end{gathered}$$

The other automorphism is the conjugation τ acting on the fiber F_x over $x \in B_n$ as follows

$$\tau : f_x \mapsto \bar{f}_x = 1 - f_x.$$

Defining the action of τ on T_λ

$$(\tau \cdot T_\lambda)(s_\lambda) = 1 - T_\lambda(\tau(s_\lambda)),$$

it follows the action of τ on ρ

$$\begin{aligned}\tau \cdot \rho &= \sum_{i_0,i_1,i_2 \in \{0,1\}} \{1 - \delta(1-i_0, 1-i_1, 1-i_2)\}\, 2^{2^2 i_0 + 2^1 i_1 + 2^0 i_2}\\ &= 255 - \varrho \cdot \rho.\end{aligned}$$

There exist 6 rules, $\rho = 240, 232, 212, 204, 178, 150$, invariant under the action of τ among the above 24 rules. There also exist 2 rules, $\rho = 226, 198$, invariant under the action of $\omega \circ \tau$ among them. Remaining $24 - 8 = 16$ rules can be identified with each other under the action of τ. Thus we may consider the following $8 + 8 = 16$ rules.

$$240,\ 232,\ 228,\ 226,\ 225,\ 212,\ 209,\ 204,\ 201,\ 198,\ 197,\ 195,\ 178,\ 165,\ 154,\ 150.$$

Let us consider the orbit decomposition of the n-WCA $\mathcal{W}(2,2)_n$ in terms of the group $G = \langle \varrho, \omega, \tau \rangle \simeq \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ generated by ϱ, ω, τ which satisfy the generating relations

$$\varrho^{,2} = \omega^2 = \tau^2 = 1, \quad \varrho\omega = \omega\varrho, \quad \varrho\tau = \tau\varrho, \quad \omega\tau = \tau\omega.$$

Define the action of G on the set of all rules in $\mathcal{W}(2,2)_n$ as above. Also define the equivalence relation $\sim$ between the rules referred to a and b as follows

$$a \sim b \qquad \Longleftrightarrow \qquad a = g \cdot b \quad \text{for } g \in G.$$

Then we obtain the quotient set $\mathcal{W}(2,2)_n/\sim$.

Now we show that the action of the group G does not affect reversibility of the rules in $\mathcal{W}(2,2)_n$.

Proposition 7. Let the group G be as above. Then a rule in the n-WCA $\mathcal{W}(2,2)_n$ referred to $\rho = g \cdot a$, where $g \in G$, is reversible if and only if the rule referred to $\rho = a$ is reversible.

(Proof) The rule $\rho = a$ in $\mathcal{W}(2,2)_n$ is reversible if and only if the global transition induced from the local transition referred to $\rho = a$ acts as a permutation on the set of all cycles of length n in $\mathcal{B}(2,2)$. Therefor, we have only to show that $g \in G$ also acts as a permutation on the set of all cycles of length n in $\mathcal{B}(2,2)$.

Since the generator ϱ of G is an automorphism on $\mathcal{B}(2,2)$[1], ϱ induces a permutation among all paths in $\mathcal{B}(2,2)$. The generator τ of G also induces the same action on $\mathcal{B}(2,2)$ as ϱ, because the action of τ on the set of all configurations is the same as ϱ:

$$\tau : s_\lambda = (f_0 f_1 f_2) \mapsto (\bar{f}_0 \bar{f}_1 \bar{f}_2),$$

where $\bar{f}_i = 1 - f_i$ for $i = 0, 1, 2$. In addition, the generator ω of G also acts on $\mathcal{B}(2,2) = (Q_0, Q_1, s, t)$ as an automorphism:

$$\omega : a_1 \leftrightarrow a_3,\ \alpha_1 \leftrightarrow \alpha_4,\ \alpha_3 \leftrightarrow \alpha_6,$$

where $Q_0 = (a_0, \cdots, a_3)$, $Q_1 = (\alpha_0, \cdots, \alpha_7)$, and other vertices and arrows are fixed[2]. Hence ω induces a permutation among all paths in $\mathcal{B}(2,2)$. This completes the proof. □

By simple observation, we see that the rule referred to the odd number ρ greater than 128 is not reversible. Therefore we have only to consider the following 10 rules

$$240,\ 232,\ 228,\ 226,\ 212,\ 204,\ 198,\ 178,\ 154,\ 150.$$

The rule referred to $\rho = 240$ is reversible because its global transition is nothing but the left shift; and the rule referred to $\rho = 204$ is also reversible because its global transition is the identity. Moreover, reversibility of the rule referred to $\rho = 150$ has already proved [24]; *i.e.,* the following proposition holds.

Proposition 8. Each rule in $\mathcal{W}(2,2)_n/\sim$ whose representative is referred to $\rho = 150$ is reversible if and only if $n \notin 3\mathbb{Z}$.

(Proof) Let $\mathcal{M}$ be the adjacency matrix of $\mathcal{W}(2,2)_n$. Let $T(\mathcal{M})$ be the transition matrix of $\mathcal{W}(2,2)_n$ whose local transition is referred to $\rho = 150$:

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (1) & (0) \\ (1) & (0) & 0 & 0 \\ 0 & 0 & (0) & (1) \end{pmatrix}.$$

Then we obtain

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (01) & (11) & (10) \\ (11) & (10) & (00) & (01) \\ (10) & (11) & (01) & (00) \\ (01) & (00) & (10) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \left(\begin{array}{cc|cc} (000)+(111) & (001)+(110) & (011)+(100) & (010)+(101) \\ (001)+(110) & (000)+(111) & (010)+(101) & (011)+(100) \\ \hline (011)+(100) & (010)+(101) & (000)+(111) & (001)+(110) \\ (010)+(101) & (011)+(100) & (001)+(110) & (000)+(111) \end{array}\right). \quad (6)$$

Let us prove the following lemma.

[1] The action of ϱ on $\mathcal{B}(2,2)$ as a planner graph is realized as the rotation by π.

[2] The action of ω on $\mathcal{B}(2,2)$ as a planner graph is realized as the reflection with respect to the line passing through the vertices a_0 and a_3.

Lemma 8. The $3m$-th power $T(\mathcal{M})^{3m}$ of the transition matrix of the n-WCA $\mathcal{W}(2,2)_n$ with the choice of the local transition referred to $\rho = 150$ is of the recursive form

$$T(\mathcal{M})^{3m} = \begin{pmatrix} M_{3m} & N_{3m} \\ N_{3m} & M_{3m} \end{pmatrix},$$

where the 2 × 2 matrices M_{3m} and N_{3m} are given as follows

$$M_{3m} = \begin{pmatrix} \mu_{3m} & \nu_{3m} \\ \nu_{3m} & \mu_{3m} \end{pmatrix}, \qquad N_{3m} = \begin{pmatrix} \xi_{3m} & \zeta_{3m} \\ \zeta_{3m} & \xi_{3m} \end{pmatrix},$$

and the entries μ_{3m}, ν_{3m}, ξ_{3m}, and ζ_{3m} are the sum of 2^{3m} different configurations of length $3m$ in $\mathcal{W}(2,2)_n$.

(Proof of lemma) By virtue of (6), the statement is true for $m = 1$. Assume it is true for $3m$. Then we have

$$\begin{aligned} T(\mathcal{M})^{3m+3} &= \begin{pmatrix} M_{3m} & N_{3m} \\ N_{3m} & M_{3m} \end{pmatrix} \begin{pmatrix} M_3 & N_3 \\ N_3 & M_3 \end{pmatrix} \\ &= \left(\begin{array}{c|c} M_{3m}M_3 + N_{3m}N_3 & M_{3m}N_3 + N_{3m}M_3 \\ \hline N_{3m}M_3 + M_{3m}N_3 & N_{3m}N_3 + M_{3m}M_3 \end{array} \right) \end{aligned}$$

It immediately follows that the statement is true for $3m + 3$. □

By virtue of lemma 8, we have

$$\begin{aligned} \mathrm{tr}\left(T(\mathcal{M})^{3m+1}\right) &= \mathrm{tr}\left(\begin{pmatrix} M_{3m} & N_{3m} \\ N_{3m} & M_{3m} \end{pmatrix} \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (1) & (0) \\ (1) & (0) & 0 & 0 \\ 0 & 0 & (0) & (1) \end{pmatrix} \right) \\ &= \mu_{3m}(0) + \xi_{3m}(1) + \nu_{3m}(1) + \zeta_{3m}(0) \\ &\quad + \zeta_{3m}(1) + \nu_{3m}(0) + \xi_{3m}(0) + \mu_{3m}(1). \end{aligned}$$

It is clear that all 2^{3m+1} terms in $\mathrm{tr}\left(T(\mathcal{M})^{3m+1}\right)$ do not coincide with each other. Therefore, the rule referred to $\rho = 150$ is reversible for $n = 3\mathbb{Z} + 1$. Similarly, we obtain

$$\begin{aligned} \mathrm{tr}\left(T(\mathcal{M})^{3m+2}\right) &= \mathrm{tr}\left(\begin{pmatrix} M_{3m} & N_{3m} \\ N_{3m} & M_{3m} \end{pmatrix} \begin{pmatrix} (00) & (01) & (11) & (10) \\ (11) & (10) & (00) & (01) \\ (10) & (11) & (01) & (00) \\ (01) & (00) & (10) & (11) \end{pmatrix} \right) \\ &= \mu_{3m}(00) + \nu_{3m}(11) + \xi_{3m}(10) + \zeta_{3m}(01) \\ &\quad + \nu_{3m}(01) + \mu_{3m}(10) + \zeta_{3m}(11) + \xi_{3m}(01) \\ &\quad + \xi_{3m}(11) + \zeta_{3m}(00) + \mu_{3m}(01) + \nu_{3m}(10) \\ &\quad + \zeta_{3m}(10) + \xi_{3m}(00) + \nu_{3m}(00) + \mu_{3m}(11), \end{aligned}$$

and hence the rule referred to $\rho = 150$ is reversible for $n = 3\mathbb{Z} + 2$. Non-reversibility for $n = 3\mathbb{Z}$ is immediate consequence from lemma 8 . □

5.2. Reversibility of Rule 154

Now we search the following 7 rules for reversible ones

$$232,\ 228,\ 226,\ 212,\ 198,\ 178,\ 154. \tag{7}$$

At first we show that the rule referred to $\rho = 154$ is reversible for $n \in 2\mathbb{Z}+1$.

Proposition 9. Each rule in $\mathcal{W}(2,2)_n/\sim$ whose representative is referred to $\rho = 154$ is reversible if and only if $n \in 2\mathbb{Z}+1$.

Remark 4. The equivalence class whose representative is $\rho = 154$ consists of the following 8 rules

$$\begin{aligned}&\omega \cdot 154 = 210, \quad \tau \cdot 154 = 166, \quad \varrho \cdot 154 = 89, \quad (\omega\tau) \cdot 154 = 180,\\ &(\varrho\omega) \cdot 154 = 75, \quad (\varrho\tau) \cdot 154 = 101, \quad (\varrho\omega\tau) \cdot 154 = 45.\end{aligned}$$

Now we study the property of the transition matrix $T(\mathcal{M})$ with the local transition referred to $\rho = 154$:

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (0) & (1) \\ (1) & (0) & 0 & 0 \\ 0 & 0 & (0) & (1) \end{pmatrix}.$$

By virtue of proposition 4, we obtain the explicit formula of the trace of the n-th power of $T(\mathcal{M})$

$$\operatorname{tr}\left(T(\mathcal{M})^n\right) = \sum_{j_0,j_1,\cdots,j_{n-2}\in\{0,1,\dots,7\}} \delta([j_0])\delta([j_1])\cdots\delta([j_{n-2}])\delta([j_{n-1}]),$$

where j_i ranges over $\{0,1,\dots,7\}$ with each successive pair (j_i, j_{i+1}) satisfying $j_{i+1} \equiv 2j_i, 2j_i+1$ (mod 8) for $i = 0,1,\dots,n-1$ and we assume $j_n = j_0$. For example, we have

$$\begin{aligned}\operatorname{tr}\left(T(\mathcal{M})^3\right) &= \sum_{j_0,j_1\in\{0,1,\dots,7\}} \delta([j_0])\delta([j_1])\delta([j_0])\\ &= \delta([0])\delta([0])\delta([0]) + \delta([1])\delta([2])\delta([4])\\ &\quad + \delta([2])\delta([4])\delta([1]) + \delta([3])\delta([6])\delta([5]) + \delta([4])\delta([1])\delta([2])\\ &\quad\quad + \delta([5])\delta([3])\delta([6]) + \delta([6])\delta([5])\delta([3]) + \delta([7])\delta([7])\delta([7])\\ &= (000) + (101) + (011) + (100) + (110) + (010) + (001) + (111).\end{aligned}$$

Lemma 9. Suppose $m \in 2\mathbb{Z}+1$ and $m \geq 2$. Let $A_m = (a_{ij}(m))$ be the m-th power $T(\mathcal{M})^m$ of the transition matrix with the local transition referred to $\rho = 154$. Then every row sum is the sum of all configurations in the n-WCA $\mathcal{W}(2,2)_n$:

$$\sum_{j=1}^{4} a_{ij}(n) = \sum_{f_0,f_1,\cdots,f_{n-1}\in\{0,1\}} (f_0 f_1 \cdots f_{n-1}) \qquad \text{for } i = 1,2,3,4.$$

(Proof) Because every row of the transition matrix $T(\mathcal{M})$ contains a (0) and a (1), all configurations of length n appear in every row of A_n just once. It is easy to understand to consider the corresponding de Bruijn quiver $\mathcal{B}(2,2)$. Let a be an arbitrary vertex in $\mathcal{B}(2,2)$. Two arrows α and β emanate from the vertex a. One of the arrow is labeled by (0) and the other by (1) this is because the fact that every row of $T(\mathcal{M})$ contains a (0) and a (1). Then the labeling of two paths from the vertex a never coincide with each other. There exist 2^m paths of length m emanating from a, and hence there exist 2^m labeling of paths each of which bijectively corresponds to a local configuration in A_m. □

Lemma 10. Suppose $m \in 2\mathbb{Z}+1$ and $m \geq 2$. Let $A_m = (a_{ij}(m))$ be as in lemma 9. Then the entries $a_{ij}(m)$ of A_m satisfy

1. $a_{13}(m) = a_{33}(m)$,

2. $a_{23}(m) = a_{43}(m)$,

3. $a_{24}(m) = a_{44}(m)$, and

4. $a_{12}(m) + a_{14}(m) = a_{22}(m) + a_{24}(m)$.

(Proof) 1. Note that the powers A_2 and A_3 of $T(\mathcal{M})$ are as follows

$$A_2 = \begin{pmatrix} (00) & (01) & (10) & (11) \\ (01) & (00) & (10) & (11) \\ (10) & (11) & (00) & (01) \\ (01) & (00) & (10) & (11) \end{pmatrix},$$

$$A_3 = \begin{pmatrix} (000)+(101) & (001)+(100) & (010)+(110) & (011)+(111) \\ (010)+(101) & (011)+(100) & (000)+(110) & (001)+(111) \\ (001)+(100) & (000)+(101) & (010)+(110) & (011)+(111) \\ (010)+(101) & (011)+(100) & (000)+(110) & (001)+(111) \end{pmatrix}.$$

It is easy to see $a_{13}(3) = a_{33}(3)$. Assume the property $a_{13}(m) = a_{33}(m)$ is true for $m \in \mathbb{Z}+1$ and $m \geq 1$. Then we have

$$\begin{aligned} a_{13}(m+2) &= \{a_{11}(m) + a_{12}(m) + a_{14}(m)\}(10) + a_{13}(m)(00) \\ &= \{c - a_{13}(m)\}(10) + a_{13}(m)(00), \end{aligned}$$

where c is the row sum $c = \sum_{i=1}^{n} a_{1j}(m)$. By the assumption $a_{13}(m) = a_{33}(m)$ of induction, we have

$$\begin{aligned} a_{13}(m+2) &= \{c - a_{33}(m)\}(10) + a_{33}(m)(00) \\ &= \{a_{31}(m) + a_{32}(m) + a_{34}(m)\}(10) + a_{33}(m)(00) \\ &= a_{33}(m+2), \end{aligned}$$

where we use the fact $c = \sum_{i=1}^{n} a_{3j}(m)$ from lemma 9.

2. The property $a_{23}(3) = a_{43}(3)$ is also true, and we assume it is true for m. Then we have

$$\begin{aligned} a_{23}(m+2) &= \{a_{21}(m) + a_{22}(m) + a_{24}(m)\}(10) + a_{23}(m)(00) \\ &= \{c - a_{23}(m)\}(10) + a_{23}(m)(00) \\ &= \{c - a_{43}(m)\}(10) + a_{43}(m)(00) \\ &= \{a_{41}(m) + a_{42}(m) + a_{44}(m)\}(10) + a_{43}(m)(00) \\ &= a_{43}(m+2). \end{aligned}$$

3. The property $a_{24}(3) = a_{44}(3)$ is also true, and we assume it is true for m. Then we have

$$\begin{aligned} a_{24}(m+2) &= \{a_{21}(m) + a_{22}(m) + a_{24}(m)\}(11) + a_{23}(m)(01) \\ &= \{c - a_{23}(m)\}(11) + a_{23}(m)(01) \\ &= \{c - a_{43}(m)\}(11) + a_{43}(m)(01) \\ &= \{a_{41}(m) + a_{42}(m) + a_{44}(m)\}(11) + a_{43}(m)(01) \\ &= a_{44}(m+2), \end{aligned}$$

where we use the fact $a_{23}(m) = a_{43}(m)$ from the statement 2.

4. The property $a_{12}(3) + a_{14}(3) = a_{22}(3) + a_{24}(3)$ is also true, and we assume it is true for m. The we have

$$\begin{aligned} & a_{12}(m+2) + a_{14}(m+2) \\ &\quad = a_{11}(m)(01) + \{a_{12}(m) + a_{14}(m)\}(00) + a_{13}(m)(11) \\ &\qquad + \{a_{11}(m) + a_{12}(m) + a_{14}(m)\}(11) + a_{13}(m)(01) \\ &\quad = \{a_{11}(m) + a_{13}(m)\}\{(01) + (11)\} + \{a_{12}(m) + a_{14}(m)\}\{(00) + (11)\} \\ &\quad = \{c - a_{12}(m) - a_{14}(m)\}\{(01) + (11)\} + \{a_{12}(m) + a_{14}(m)\}\{(00) + (11)\} \\ &\quad = \{c - a_{22}(m) - a_{24}(m)\}\{(01) + (11)\} + \{a_{22}(m) + a_{24}(m)\}\{(00) + (11)\} \\ &\quad = \{a_{21}(m) + a_{23}(m)\}\{(01) + (11)\} + \{a_{22}(m) + a_{24}(m)\}\{(00) + (11)\} \\ &\quad = a_{22}(m+2) + a_{24}(m+2) \end{aligned}$$

Thus we finish the proof. □

By using the above two lemmas, we can easily prove proposition 9.

(Proof of proposition 9) Assume $n \in 2\mathbb{Z} + 1$. Then, from lemma 10, we have

$$\begin{aligned} \mathrm{tr}\, A_n &= a_{11}(n) + a_{22}(n) + a_{33}(n) + a_{44}(n) \\ &= a_{11}(n) + a_{22}(n) + a_{13}(n) + a_{24}(n) \\ &= a_{11}(n) + a_{12}(n) + a_{13}(n) + a_{14}(n). \end{aligned}$$

This is nothing but the total sum of the entries in the first row of A_n. From lemma 9, it follows that

$$\mathrm{tr}\, A_n = \sum_{f_0, f_1, \cdots, f_{n-1} \in \{0,1\}} (f_0 f_1 \cdots f_{n-1}) = \mathrm{tr}\left(\mathcal{M}^n\right),$$

where $\mathcal{M}$ is the adjacency matrix of the n-WCA $(2,2)_n$. Thus each rule in $\mathcal{W}(2,2)_n/\sim$ represented by $\rho = 154$ is reversible for $n \in 2\mathbb{Z}+1$ (see theorem 2).

Assume $n \in 2\mathbb{Z}$ and consider

$$A_2 = \begin{pmatrix} (00) & (01) & (10) & (11) \\ (01) & (00) & (10) & (11) \\ (10) & (11) & (00) & (01) \\ (01) & (00) & (10) & (11) \end{pmatrix}.$$

Since the $(1,1)$, $(2,2)$, and $(3,3)$-entries of A_2 are all (00), the same entries in A_{2m} contain the configuration $(0000\cdots00)$ of length $2m$ for any $m \geq 1$, respectively. Therefore we have

$$\text{tr } A_{2m} \neq \sum_{f_0,f_1,\cdots,f_{2m-1}\in\{0,1\}} (f_0 f_1 \cdots f_{2m-1}) = \text{tr}\left(\mathcal{M}^{2m}\right)$$

This completes the proof of proposition 9. □

5.3. Complete List of Reversible Rules

Next we show that the remaining 6 rules referred to $\rho = 232, 228, 226, 212, 198, 178$ are not reversible. Non-reversibility of the n-WCA $\mathcal{W}(l,r)_n$ is easily checked in terms of theorem 4 concerning the transition matrix $T(\mathcal{M})$.

Let us consider the transition matrix $T(\mathcal{M})$ with the local transition referred to $\rho = 232$:

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (0) & 0 & 0 \\ 0 & 0 & (0) & (1) \\ (0) & (1) & 0 & 0 \\ 0 & 0 & (1) & (1) \end{pmatrix}.$$

Then we have

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (00) & (00) & (01) \\ (00) & (01) & (11) & (11) \\ (00) & (00) & (10) & (11) \\ (10) & (11) & (11) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \begin{pmatrix} 2(000) & (000)+(001) & (000)+(011) & (001)+(011) \\ (000)+(110) & (000)+(111) & (010)+(111) & (011)+(111) \\ (000)+(100) & (000)+(101) & (000)+(111) & (001)+(111) \\ (100)+(110) & (100)+(111) & (110)+(111) & 2(111) \end{pmatrix}.$$

We see that the $(1,1)$-entry of $T(\mathcal{M})^3$ is $2(000)$. Therefore, by virtue of theorem 4, each rule in the equivalence class represented by $\rho = 232$ is not reversible for $n \geq 3$.

Similarly, for the rule referred to $\rho = 228$, we obtain

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (0) & 0 & 0 \\ 0 & 0 & (1) & (0) \\ (0) & (1) & 0 & 0 \\ 0 & 0 & (1) & (1) \end{pmatrix},$$

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (00) & (01) & (00) \\ (10) & (11) & (01) & (01) \\ (00) & (00) & (11) & (10) \\ (10) & (11) & (11) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \begin{pmatrix} (000)+(010) & (000)+(011) & 2(001) & (000)+(001) \\ (100)+(010) & (100)+(011) & (111)+(011) & (110)+(011) \\ (000)+(110) & (000)+(111) & (001)+(101) & (000)+(101) \\ (100)+(110) & (100)+(111) & 2(111) & (110)+(111) \end{pmatrix}.$$

The $(1,3)$-entry of $T(\mathcal{M})^3$ is $2(001)$, therefore each rule in the equivalence class represented by $\rho = 228$ is not reversible for $n \geq 4$.

For the rule referred to $\rho = 226$, we obtain

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (0) & (0) \\ (0) & (1) & 0 & 0 \\ 0 & 0 & (1) & (1) \end{pmatrix},$$

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (01) & (10) & (10) \\ (00) & (01) & (01) & (01) \\ (00) & (01) & (10) & (10) \\ (10) & (11) & (11) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \begin{pmatrix} (000)+(100) & (001)+(101) & (010)+(101) & (010)+(101) \\ (000)+(010) & (001)+(011) & (010)+(011) & (010)+(011) \\ (000)+(100) & (001)+(101) & (010)+(101) & (010)+(101) \\ (100)+(110) & (101)+(111) & (110)+(111) & (110)+(111) \end{pmatrix}.$$

Then the $(2,2)$-entry of $T(\mathcal{M})^4$ is

$$(0001) + 2(0101) + (0111),$$

and hence each rule in the equivalence class represented by $\rho = 226$ is not reversible for $n \geq 4$.

Proceed further. For the rule referred to $\rho = 212$, we obtain

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (0) & 0 & 0 \\ 0 & 0 & (1) & (0) \\ (1) & (0) & 0 & 0 \\ 0 & 0 & (1) & (1) \end{pmatrix},$$

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (00) & (01) & (00) \\ (11) & (10) & (01) & (01) \\ (10) & (10) & (01) & (00) \\ (11) & (10) & (11) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \begin{pmatrix} (000)+(011) & (000)+(010) & 2(001) & (000)+(001) \\ (110)+(011) & (110)+(010) & (101)+(011) & (100)+(011) \\ (100)+(011) & (100)+(010) & (101)+(001) & (100)+(001) \\ (110)+(111) & 2(110) & (101)+(111) & (100)+(111) \end{pmatrix}.$$

Therefore each rule in the equivalence class represented by $\rho = 212$ is not reversible for $n \geq 4$.

For the rule referred to $\rho = 198$, we obtain

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (1) & (0) \\ (0) & (0) & 0 & 0 \\ 0 & 0 & (1) & (1) \end{pmatrix},$$

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (01) & (11) & (10) \\ (10) & (10) & (01) & (01) \\ (00) & (01) & (01) & (00) \\ (10) & (10) & (11) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \begin{pmatrix} (000)+(110) & (001)+(110) & (011)+(101) & (010)+(101) \\ (100)+(010) & (101)+(010) & (101)+(011) & (100)+(011) \\ (000)+(010) & (001)+(010) & (011)+(001) & (010)+(001) \\ (100)+(110) & (101)+(110) & (101)+(111) & (100)+(111) \end{pmatrix}.$$

Then the $(3,3)$-entry of $T(\mathcal{M})^4$ is

$$(0011) + 2(0101) + (0011),$$

and hence each rule in the equivalence class represented by $\rho = 198$ is not reversible for $n \geq 4$.

Finally, for the rule referred to $\rho = 178$, we obtain

$$T(\mathcal{M}) = \begin{pmatrix} (0) & (1) & 0 & 0 \\ 0 & 0 & (0) & (0) \\ (1) & (1) & 0 & 0 \\ 0 & 0 & (0) & (1) \end{pmatrix},$$

$$T(\mathcal{M})^2 = \begin{pmatrix} (00) & (01) & (10) & (10) \\ (01) & (01) & (00) & (01) \\ (10) & (11) & (10) & (10) \\ (01) & (01) & (10) & (11) \end{pmatrix},$$

$$T(\mathcal{M})^3 = \begin{pmatrix} (000)+(101) & (001)+(101) & (010)+(100) & (010)+(101) \\ (010)+(001) & (011)+(001) & 2(010) & (010)+(011) \\ (100)+(101) & 2(101) & (110)+(100) & (110)+(101) \\ (010)+(101) & (011)+(101) & (010)+(110) & (010)+(111) \end{pmatrix}.$$

Therefore each rule in the equivalence class represented by $\rho = 178$ is not reversible for $n \geq 4$.

Summarizing the above facts, we finally obtain the following theorem concerning reversibility of ECA imposing periodic boundary conditions.

Theorem 5. There exist exactly 16 reversible rules in the n-WCA $\mathcal{W}(2,2)_n$ which are referred to the numbers in the following table.

Representative	ω	τ	ϱ	$\omega\tau$	$\varrho\omega$	$\varrho\tau$	$\varrho\omega\tau$	Period
150			105					$n \in 3\mathbb{Z}+1, 3\mathbb{Z}+2$
154	210	166	89	180	75	101	45	$n \in 2\mathbb{Z}+1$
170	240		85		15			$n \in \mathbb{N}$
204			51					$n \in \mathbb{N}$

In the above table, each element of $G = \langle \varrho, \omega, \tau \rangle$ maps the representative of $\mathcal{W}(2,2)_n/\sim$ into a rule below it. The rules are reversible only for the period n of the base space listed in the rightmost column. □

Remark 5 (see [35]). The rule referred to $\rho = 154$ has the property called linearizability. Therefore, we can obtain a formula computing the period with respect to the global transition for any initial configuration. Moreover, we can prove that there exist infinitely many linearizable n-WCA $\mathcal{W}(l,r)_n$; we can also obtain a formula computing the period with respect to its global transition for any initial configuration.

6. Conclusion

We establish a on-to-one correspondence between the configurations in the WCA, which is a family consisting of infinitely many cellular automata, and the paths in the de Bruijn quiver, which is also a family consisting of infinitely many quivers. Extending the correspondence to that between the configuration algebra of the WCA and the path algebra of the de Bruijn quiver both of which are associative algebras, we obtain the global transition of the configuration algebra of the WCA. Thus we translate the problem concerning reversibility of the WCA into that concerning surjectivity of the endomorphism, which is induced from the local transition of the WCA, on the configuration algebra of the WCA. We then show that the induced problem concerning the endomorphism can be solved in terms of the adjacency matrix of the WCA, which is defined from that of the de Bruijn quiver through the one-to-one correspondence. Indeed, we give a necessary and sufficient condition for reversibility of the WCA. By virtue of the necessary and sufficient condition, we classify all 16 reversible rules in the ECA imposing periodic boundary conditions.

As we have referred to in section 1., we search the WCA for reversible cellular automata, and we consequently obtain several reversible rules in the ECA. Although we could not refer to here, there exists a family consisting of infinitely many reversible rules in the WCA called the linearizable cellular automata and abbreviated to the LCA [35]. The most simple example of the LCA is nothing but the reversible ECA referred to rule 154. Since the initial value problem for each member of the LCA imposing periodic boundary conditions can be solved, *i.e.*, a formula computing the period with respect to the global transition for arbitrary initial configuration is obtained, we believe that each member of the LCA is a candidate for integrable cellular automaton. In order to establish the integrability of the LCA, to precisely study its property is a further problem. The method which associates a family of cellular automata to an associative algebra is a powerful tool in order to examine the property of the family of cellular automata. The authors think that the quantum cellular automata [27] can also be studied by using the method we have developed here. We shall report on this subject in a forth coming paper.

References

[1] Amoroso S and Patt Y, “Decision procedures for surjectivity and injectivity of parallel maps for tesselation structures”, *J. Computer and System Sciences* **6** (1972) 448-464.

[2] Assem I, Simson D and Skowronski A, *Elements of the Representation Theory of Associative Algebras Volume 1 Techniques of Representation Theory: London Mathematica Society Student Text* **207** (Cambridge: Cambridge University Press) (2006).

[3] Bialecki M, “Integrable 1D Toda cellular automata”, *J. Nonlinear Math. Phys.* **12** Supplement 2 (2005) 28?35.

[4] Fukuda K, Okado M and Yamada M, “Energy functions in box ball systems”, *Internat. J. Modern Phys.* A **15** (2000) 1379-1392.

[5] Godsil C and Royle G, *Algebraic Graph Theory: Graduate Texts in Mathematics* **207** (New York: Springer-Verlag) (2001).

[6] Hatayama G, Hikami K, Inoue R, Kuniba A, Takagi T and Tokihiro T, “The $A_M^{(1)}$ automata related to crystals of symmetric tensors”, *J. Math. Phys.* **42** (2001) 274-308.

[7] Inoue R and Takenawa T, “Tropical spectral curves and integrable cellular automata”, *Int. Math. Res. Not.* **2008** Art. ID rnn019 (2008) 27.

[8] Inoue R and Takenawa T, “A tropical analogue of Fay’s trisecant identity and the ultradiscrete periodic Toda lattice”, *Preprint* arXiv:0806.3318 (2008).

[9] Idzumi M, Iwao S, Mada J and Tokihiro T, “Solution to the initial value problem of the ultradiscrete periodic Toda equation”, *J. Phys. A: Math. Theor.* **42** (2009) 315209.

[10] Isojima S,Murata M, Nobe A and Satsuma J, “An ultradiscretization of the sine-Gordon equation”, *Phys. Lett.* A **331** (2004) 378-386.

[11] Iwao S and Tokihiro T, ”Ultradiscretization of the theta function solution of pd Toda”, *J. Phys. A: Math. Theor.* **40** (2007) 12987-13021.

[12] Iwao S, “Integration over tropical plane curves and ultradiscretization”, *Int. Math. Res. Not.* **2010** no. 1 (2010) 112-148.

[13] Jen E, “Scaling of Preimages in Cellular Automata”, *Complex Systems* **1** (1987) 1045-1062.

[14] Kanai M, Nishinari K and Tokihiro T, “Stochastic optimal velocity model and its long-lived metastability”, *Phys. Rev.* E **72** (2005) 035102-5(R).

[15] Kanai M, Nishinari K and Tokihiro T, “Analytical study on the criticality of the Stochastic Optimal Velocity model”, *J. Phys. A: Math. Gen.* **39** (2006) 2921-2933.

[16] Kanai M, Nishinari K and Tokihiro T, “Exact solution and asymptotic behaviour of the asymmetric simple exclusion process on a ring”, *J. Phys. A: Math. Gen.* **39** (2006) 9071-9079.

[17] Kearney M J, “Exactly solvable cellular automaton traffic jam model”, *Phys. Rev.* E **74** (2006) 061115.

[18] Kuniba A, Takagi T and Takenouchi A, “Bethe ansatz and inverse scattering transform in a periodic box-ball system”, *Nucl. Phys.* **B747** [PM] (2006) 354-397.

[19] Kuniba A and Sakamoto R, “The Bethe ansatz in a periodic box-ball system and the ultradiscrete Riemann theta function”, *J. Stat. Mech.* **2006** (2006) P09005.

[20] Kuniba A and Sakamoto R, “Combinatorial Bethe ansatz and generalized periodic box-ball system”, *Reviews. Math. Phys.* **20** (2008) 493-527.

[21] Mada J, Idzumi M and Tokihiro T, ”Fundamental cycle of a periodic box-ball system and solvable latticemodels”, *J. Phys. A: Math. Gen.* **39** (2006) 4985-4997.

[22] Mada J, Idzumi M and Tokihiro T, “The box-ball system and the N-soliton solution of the ultradiscrete KdV equation”, *J. Phys. A: Math. Theor.* **41** (2008) 175207.

[23] Mada J and Tokihiro T, “Correlation functions for a periodic box-ball system”, *J. Phys. A: Math. Theor.* **43** (2010) 135205.

[24] Martin O, Odlyzko A and Wolfram S, “Algebiaic properties of cellular automata”, *Comm. Math. Phys.* **93** (1984) 219-259.

[25] Matsukidaira J, Satsuma J, Takahashi D, Tokihiro T and Torii M, “Toda-type Cellular Automaton and its N-soliton solution”, *Phys. Lett.* A **225** (1997) 287-295.

[26] Mendelsohn N S, “Directed graphs with unique path property”, *Combinatorial Theory and its Applications II Proc. Colloq. Balatonfüred, 1969* (North-Holland, Amsterdam) (1970) 783-799.

[27] Meyer D A, “From quantum cellular automata to quantum lattice gases”, *J. Statist. Phys.* **85** (1996) no. 5-6 551?574.

[28] Nagai A, Tokihiro T, Satsuma J, Willox R and Kajiwara K, “Two-dimensional soliton cellular automaton of deautonomized Toda-type”, *Preprint* arXiv:solv-int/9708001 (1997).

[29] Nagai A, Takahashi D and Tokihiro T, “Soliton Cellular Automaton, Toda Molecule Equation and Sorting Algorithm”, *Phys. Lett.* A **255** (1999) 265-271.

[30] Nagai H, “A New Expression of Soliton Solution to the Ultradiscrete Toda Equation”, *J. Phys. A: Math. Theor.* **41** (2008) 235204.

[31] Nasu M, “Local Maps Inducing Surjective Global Maps of One Dimensional Tessellation Automata” *Math. Systems Theory* **11** 327-351 (1978).

[32] Nishinari K and Takahashi D, “Analytical Properties of Ultradiscrete Burgers Equation and Rule-184 Cellular Automaton”, *J. Phys. A: Math. Gen.* **31** (1998) 5439-5450.

[33] Nishinari K and Takahashi D, “A new deterministic CA model for traffic flow with multiple states”, *J. Phys. A: Math. Gen.* **32** (1998) 93-104.

[34] Nobe A and Yura F, “On reversibility of cellular automata with periodic boundary conditions”, *J. Phys. A: Math. Gen.* (2004) **37** 5789-5804.

[35] Nobe A and Yura F, “Linearizable cellular automata”, *J. Phys. A: Math. Gen.* (2007) **40** 7159-7174.

[36] Takahashi D and Matsukidaira J, “Box and Ball System with a Carrier and Ultradiscrete Modified KdV Equation”, *J. Phys. A: Math. Gen.* **30** (1997) L733-L739.

[37] Takahashi D and Satsuma J, “A Soliton Cellular Automaton”, *J. Phys. Soc. Japan* **59** (1990) 3514-3519.

[38] Toffoli T and Margolus N, “Invertible cellular automata: a review”, *Physica* D **45** (1990) 229-253.

[39] Tokihiro T and Mada JC ”Fundamental cycle of a periodic box-ball systems: a number theoretical aspect”, *Glasgow Math. J.* **47A** (2005) 199-204.

[40] Tokihiro T, Takahashi D, Matsukidaira J and Satsuma J, “From Soliton Equations to Integrable Cellular Automata through a Limiting Procedure”, *Phys. Rev. Lett.* **76** (1996) 3247-3250.

[41] Torii M, Takahashi D and Satsuma J, “Combinatorial Representation of Invariants of a Soliton Cellular Automaton”, *Physica* D **92** (1996) 209-220.

[42] Wolfram S, “Statistical mechanics of cellular automata”, *Rev. Mod. Phys.* **55** (1983) 601-644.

[43] Wolfram S, “Universarity and complexity in cellular automata”, *Physica* D **10** (1984) 1-35.

[44] Wolfram S, “Twenty Problems in the Theory of Cellular Automata”, *Phys. Scr.* T **9** (1985) 170-183.

[45] Yoshihara D, Yura F and Tokihiro T, “Fundamental Cycle of a Periodic Box-Ball System”, *J. Phys. A: Math. Gen.* **36** (2003) 99-121.

[46] Yura F and Tokihiro T, “On a periodic soliton cellular automaton”, *J. Phys. A: Math. Gen.* **35** (2002) 3787-3801.

In: Cellular Automata
Editor: Thomas M. Li, pp. 211-247

ISBN 978-1-61761-592-4

Chapter 9

FROM GLIDERS TO UNIVERSALITY OF CELLULAR AUTOMATA: ANOTHER 2D 2-STATE UNIVERSAL AUTOMATON

Emmanuel Sapin[*]
CNED

Abstract

This paper deals with the emergence of computation in complex systems. The Turing universality (i.e. the ability to encompass the whole computation power of the class of Turing machines) of cellular automata which are the simplest representation of complex systems is considered.

We aim to construct an automatic system for the discovery of Turing-universal cellular automata. In this chapter, some steps towards this objective are presented, as is the search for self-localized patterns of non-resting states called gliders. An evolutionary method to search for gliders, based on a specific fitness functions taking into account the presence of periodic patterns and gliders, led to the discovery of a large number of gliders. Among the automata accepting gliders that were discovered, some would surprisingly generate glider guns for nearly every evolution of a random cell configuration. The first of them that was discovered is picked up as a potential candidate for a universal automaton.

Patterns that are able to stop unwanted streams of gliders are called eaters, they are searched for and used with gliders and glider guns to demonstrate the universality of an automaton.

PACS 05.45-a, 52.35.Mw, 96.50.Fm.

Keywords: Cellular Automata, Glider, Glider Gun, AND.

1. Introduction

The theories of complexity are the understanding of how independent agents are interacting in a system to influence each other and the whole system [35]. A complex system can be

[*] E-mail address: emmanuelsapin@hotmail.com

described as a system composed of interconnected parts in which the whole exhibits more properties that the sum of the parts [41, 42, 43]. Surprising computational tasks could result from interactions of independent agents in complex systems as emergence of computation is a hot topic in the science of complexity [17]. A promising environment to study emergent computation is cellular automata [18] which are the simplest mathematical representation of complex systems [36] and an important modelling paradigm in the natural sciences and an extremely useful approach in the study of complex systems [44]. They are uniform frameworks in which the simple agents are cells evolving through time on the basis of a local function, called the transition rules [1].

Emerging computation in cellular automata has different forms. Some have studied specific computation like density and synchronization tasks [15, 16, 21, 25, 20] and pattern recognition [33]. While others have considered *Turing-universal automata* [2, 10, 11, 12, 13, 39, 45] i.e. automata encompassing the whole computational power of the class of Turing machines [37]. Some have asked the question of the frequency of universal cellular automata as Wolfram [3]. In order to find universal automata, we aim to construct an automatic system for the discovery of Turing-universal cellular automata.

The first 2D 2-state automaton proved Turing-universal was the Game of Life of Conway *et al.* [6]. Its demonstration of universality uses the presence of mobile self-localized patterns of non-resting states [39], called *gliders* and their generators called *glider guns* which, when evolving alone, periodically recover their original shape after emitting a number of gliders. Considering the ability of gliders to lead to universality, one of the possible first steps towards an automatic system for the discovery of Turing-universal automata is the search for gliders presented in this chapter.

The search for gliders was notably explored by Adamatzky *et al.* with a phenomenological search [22], Wuensche who used his Z-parameter and entropy [23] and Eppstein [24]. Lohn *et al.* [27], Ventrella [28] has searched for gliders using stochastic algorithms. Here, a search for gliders by evolutionary algorithms is described and the discovered gliders are described thanks to a classification taking into account periods and velocities.

This search for gliders allows the discovery of an automaton called R with glider guns: during the evolution of a random configuration of cells by this automaton glider guns emerge. This chapter presents a demonstration of the Turing universality of R.

After a second section that introduces some formalisations and notations, Section 3 presents the Game of Life while the search for gliders is described in Section 4. Section 5 deals with the demonstration of universality of R. The last section summarizes the presented results and discusses directions for future research.

2. Formalisations and Notations

2.1. Set of Cellular Automata

A cellular automaton of dimension d is a 4-tuple ($\mathbb{Z}^d$,S,V,$\delta : S^{n+1} \to S$) where :

- S is a finite set of states.

- V is a finite ordered subset of $\mathbb{Z}^d$ of cardinal n called a neighbourhood.

- $\delta : S^{n+1} \rightarrow S$ is the *local transition rule* of the cellular automaton.

In the following, we will consider only the automata with the following characteristics:

- $d = 2$.
- $S = \{0,1\}$
- V will be the Moore neighbourhood i.e. the eight direct neighbours of the element (x,y) called *cell*, therefore $n = 8$.

The space $\{(\mathbb{Z}^2,\{0,1\},\text{Moore neighbourhood},\delta),\forall\delta : S^9 \rightarrow S\}$ of these automata is called the space $\mathcal{E}$. The next formalisations will be given for automata of $\mathcal{E}$ but could be generalized for automata with higher dimensions and more states.

2.2. Evolution of Cellular Automata

A configuration at time $t \in \mathbb{N}$, or generation t, of an automaton $A \in \mathcal{E}$ is an application $c_t^A : \mathbb{Z}^2 \rightarrow \{0,1\}$. The image $c_t^A(x,y)$ is called the state of the cell (x,y). Cells in states 0 and 1 are respectively called dead and alive. The sequence $(c_t^A)_{t\geq 0}$ is said the evolution of the cellular automaton A from the configuration c_0^A if and only if:

$$\begin{aligned} c_{t+1}^A(x,y) = \delta\big(& c_t^A(x+1,y-1), c_t^A(x+1,y), c_t^A(x+1,y+1), c_t^A(x,y-1),\\ & c_t^A(x,y), c_t^A(x,y+1), c_t^A(x-1,y-1), c_t^A(x-1,y), c_t^A(x-1,y+1)\big). \end{aligned}$$

For readability, in the following, the arguments of the function δ will appear in this disposition:

$$\begin{aligned} c_{t+1}^A(x,y) = \delta\big(c_t^A(x+1,y-1), \quad & c_t^A(x+1,y), \quad c_t^A(x+1,y+1),\\ c_t^A(x,y-1), \quad & c_t^A(x,y), \quad c_t^A(x,y+1),\\ c_t^A(x-1,y-1), \quad & c_t^A(x-1,y), \quad c_t^A(x-1,y+1)\big). \end{aligned}$$

The function δ determines what will become to a cell at the next generation depending on its neighbourhood.

Figure 1, in which cells in states 0 and 1 are represented respectively by white and black coloured squares, shows the first four generations of two sequences $(c_t^A)_{t\geq 0}$, only the second one is the evolution an automaton.

2.3. Isotropy

An automaton $(\mathbb{Z}^2,\{0,1\},\text{Moore neighbourhood},\delta : S^9 \rightarrow S)$ is said *isotropic* if and only if for all binary numbers $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8$ and x_9 we get:

$$\delta\begin{pmatrix} x_1,x_2,x_3,\\ x_4,x_5,x_6,\\ x_7,x_8,x_9 \end{pmatrix} = \delta\begin{pmatrix} x_3,x_2,x_1,\\ x_6,x_5,x_4,\\ x_9,x_8,x_7 \end{pmatrix} = \delta\begin{pmatrix} x_7,x_8,x_9,\\ x_4,x_5,x_6,\\ x_1,x_2,x_3 \end{pmatrix} = \delta\begin{pmatrix} x_1,x_4,x_7,\\ x_2,x_5,x_8,\\ x_3,x_6,x_9 \end{pmatrix} =$$

$$\delta\begin{pmatrix} x_9,x_6,x_3,\\ x_8,x_5,x_2,\\ x_7,x_4,x_1 \end{pmatrix} = \delta\begin{pmatrix} x_7,x_4,x_1,\\ x_8,x_5,x_2,\\ x_9,x_6,x_3 \end{pmatrix} = \delta\begin{pmatrix} x_9,x_8,x_7,\\ x_6,x_5,x_4,\\ x_3,x_2,x_1 \end{pmatrix} = \delta\begin{pmatrix} x_3,x_6,x_9,\\ x_2,x_5,x_8,\\ x_1,x_4,x_7 \end{pmatrix}.$$

The subset of isotropic automata is called the space $I \subset \mathcal{E}$.

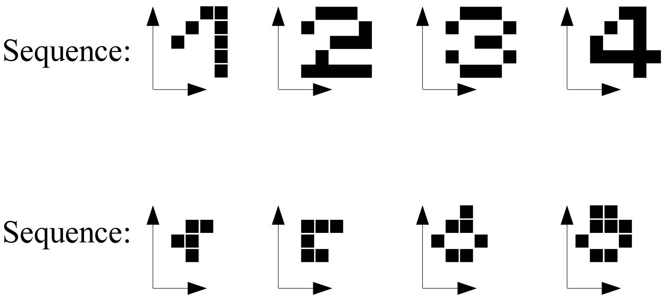


Figure 1. Two sequences $(c_t^A)_{t\geq 0}$ for t equals 0, 1, 2 and 3. Only the sequence at the bottom is the evolution of a cellular automaton that is the R-pentomino of the Game of Life [6]. States 0 and 1 are shown respectively by white and black coloured squares.

2.4. Number of Automata

The number of automata in $\mathcal{E}$ is the number of possible transition rules $\delta : S^{8+1} \rightarrow S$. There are $2^9 = 512$ different rectangular 9-cell neighbourhood states (including the central cell) therefore the space $\mathcal{E}$ contains 2^{512} automata. An automaton of $\mathcal{E}$ can be described by telling what will be the new state of a cell at the next generation depending on its neighbourhood as shown figure 2.

The number of automata of I depends on in how many subsets of isotropic neighbourhood states the 512 different rectangular 9-cell neighbourhood states can be put. Let:

$$\begin{pmatrix} x_1, x_2, x_3, \\ x_4, x_5, x_6, \\ x_7, x_8, x_9 \end{pmatrix}$$

be a rectangular 9-cell neighbourhood states. Its isotropic neighbourhood states are :

$$\begin{pmatrix} x_3, x_2, x_1, \\ x_6, x_5, x_4, \\ x_9, x_8, x_7 \end{pmatrix}, \begin{pmatrix} x_7, x_8, x_9, \\ x_4, x_5, x_6, \\ x_1, x_2, x_3 \end{pmatrix}, \begin{pmatrix} x_1, x_4, x_7, \\ x_2, x_5, x_8, \\ x_3, x_6, x_9 \end{pmatrix}, \begin{pmatrix} x_9, x_6, x_3, \\ x_8, x_5, x_2, \\ x_7, x_4, x_1 \end{pmatrix}, \begin{pmatrix} x_7, x_4, x_1, \\ x_8, x_5, x_2, \\ x_9, x_6, x_3 \end{pmatrix},$$

$$\begin{pmatrix} x_9, x_8, x_7, \\ x_6, x_5, x_4, \\ x_3, x_2, x_1 \end{pmatrix}, \begin{pmatrix} x_3, x_6, x_9, \\ x_2, x_5, x_8, \\ x_1, x_4, x_7 \end{pmatrix}.$$

An exhaustive study shows that, depending on the values of the nine binary numbers $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8$ and x_9, the eight isotropic neighbourhood states could be:

- all different from one another or
- equal to one another or
- equal two-by-two or
- equal four-by-four.

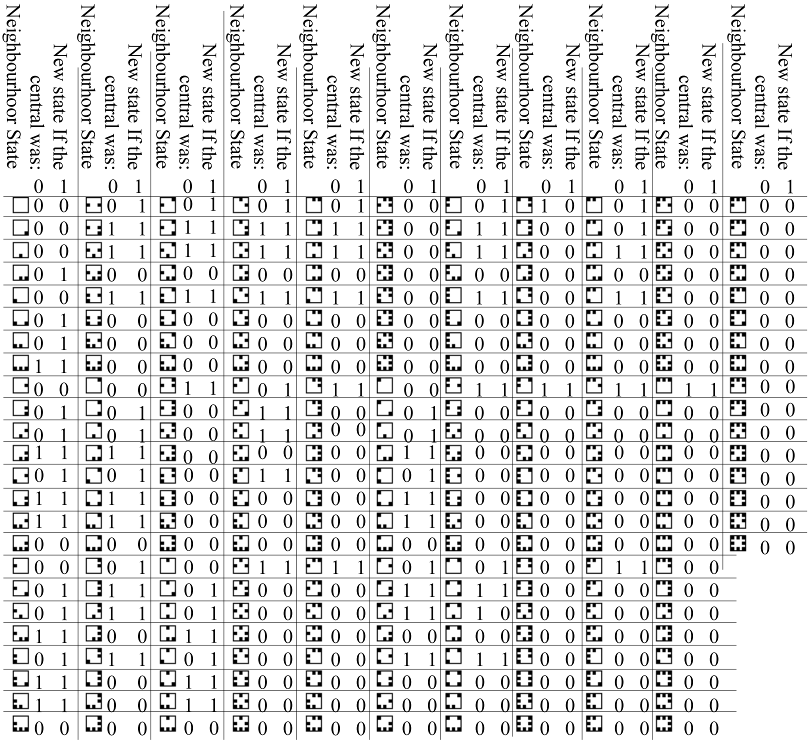


Figure 2. An automaton of the space $\mathcal{E}$.

That leads to having subsets of isotropic neighbourhood states with a different number of elements. Figures 3, 4, 5, 6 show subsets of 1, 2, 4 and 8 elements in which cells in states 0 and 1 are represented respectively by white and black coloured squares. These figures show the subsets of isotropic neighbourhood states in which $x_5 = 1$.

All the 512 different rectangular 9-cell neighbourhood states can be put in 102 subsets of isotropic neighbourhood states, meaning that there are 2^{102} different automata in I. In order to describe an automaton of I, one just needs to be able to tell what the state of a cell will be at the next generation, depending on which subset of isotropic neighbourhood states its neighbourhood is in. The subset of isotropic neighbourhood states is represented by one element on figure 7 that shows an automaton of I.

2.5. Quiescent State

Among the states of a cellular automaton, sometime, a state s, called *quiescent state* [38], is such that:

$$\delta(s, \ldots, s) = s$$

In the following the state 0 will be the quiescent state. There are 2^{511} and 2^{101} automata of $\mathcal{E}$ and I for which 0 is the quiescent state.

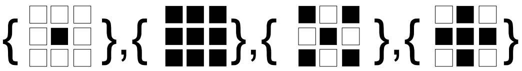


Figure 3. The four subsets of one isotropic neighbourhood state in which x_5=1. The first subset is composed with the element (0,0,0,0,1,0,0,0,0) as states 0 and 1 are shown respectively by white and black coloured squares.

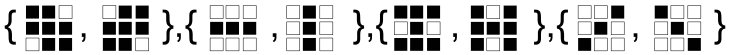


Figure 4. The four subsets of two isotropic neighbourhood states.

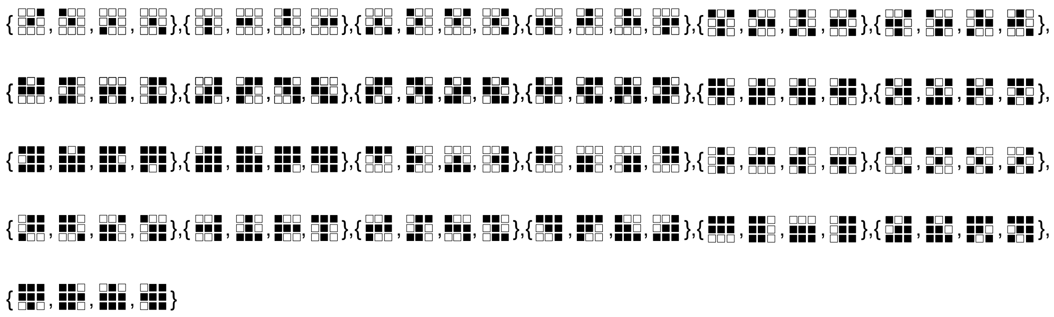


Figure 5. The twenty five subsets of four isotropic neighbourhood states.

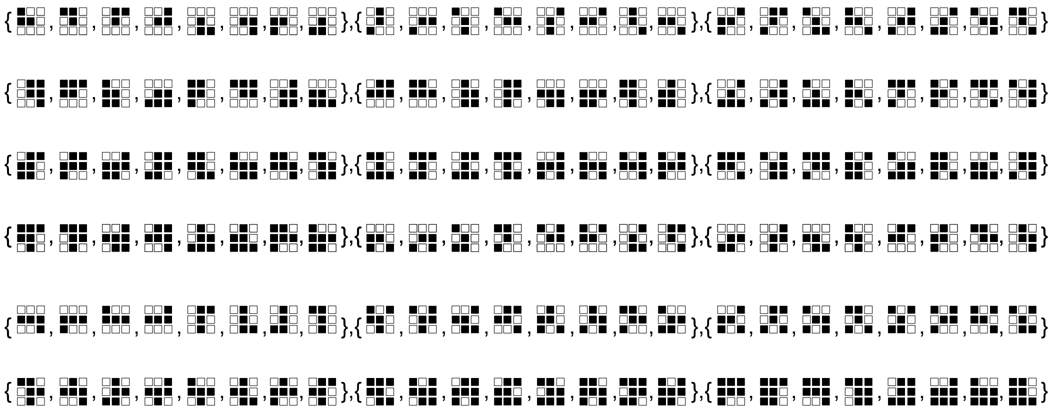


Figure 6. The eighteen subsets of eight isotropic neighbourhood states.

Neighbourhood State	New state If the central was:		Neighbourhood State	New state If the central was:		Neighbourhood State	New state If the central was:		Neighbourhood State	New state If the central was:		Neighbourhood State	New state If the central was:		Neighbourhood State	New state If the central was:		Neighbourhood State	New state If the central was:	
	0	1		0	1		0	1		0	1		0	1		0	1		0	1
	0	0		0	0		0	1		0	1		0	1		0	0		0	1
	0	0		0	0		1	1		1	1		1	1		0	0		1	1
	0	0		0	0		1	1		1	1		1	1		0	0		1	1
	0	1		0	1		0	0		0	0		0	0		0	0			
	0	0		0	0		1	1		1	1		1	1		0	0			
	0	1		0	1		0	0		0	0		0	0		0	0			
	0	1		0	1		0	0		0	0		0	0		0	0			
	1	1		1	1		0	0		0	0		0	0		0	0			

Figure 7. An automaton of the space I.

2.6. Patterns

2.6.1. Definition

A *pattern* P is an application $R \rightarrow S$, with R a given rectangular subset of $\mathbb{Z}^2$ and S equal to $\{0,1\}$.

A pattern is included in a configuration c_t^A of an automaton A of the space $\mathcal{E}$ if the restriction of c_t^A to R (or a translation of R in $\mathbb{Z}^2$) is equal to P. A pattern is included in a configuration c_t^A of an isotropic automaton A of the space I if the restriction of c_t^A to R (or composition of a translation and symetries of R in $\mathbb{Z}^2$) is equal to P. Figure 8 shows the R-pentomino of Game of Life after applying symetries.

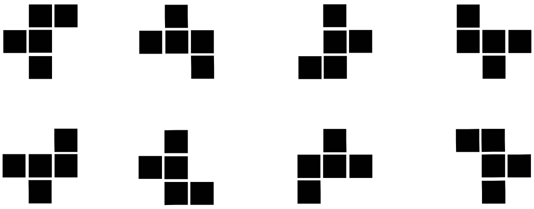


Figure 8. The R-pentomino of Game of Life after applying symetries.

The sequence $(P_t^A)_{t\geq 0}$ is an evolution of P in the automaton A if P is included in the configuration P_0^A and P_0^A is zero outside of the associated rectangle of $\mathbb{Z}^2$.

2.6.2. Glider

A *glider* G of the automaton A is a finite cycle of patterns $[P_0, P_1, ..., P_{T-1}]$ such that:

- If $(P_t^A)_{t\geq 0}$ is an evolution of P_0 in the automaton A, then $(P_{t+i}^A)_{t\geq 0}$ is an evolution of P_i in the automaton A, for $i = 1$ to $T-1$.

- There exists two integers D_x and D_y such that $(x,y) \neq (0,0)$ and for all integers x and y we get:
$$G_T^A(x,y) = G_0^A(x+D_x, y+D_y).$$

The number T is called the period of the glider.

D_x and D_y are respectively the deplacements along X-axis and Y-axis. The movement is how much the glider moves during a period. The definition of the movement depends on the chosen neighbourhood. Considering Moore neighbourhood, the movement is defined by the maximum between $|D_x|$ and $|D_y|$.

The definition of the velocity S of a glider is the movement divides by the period T.

A glider can be *orthogonal*, *diagonal* or *oblique*:

- If D_x equals 0 or D_y equals 0, G is said an orthogonal glider.
- If $|D_x|$ equals $|D_y|$, G is said a diagonal glider.
- If G is neither diagonal nor orthogonal, G is said an oblique glider.

The figure 9 shows orthogonal and diagonal gliders of the Game of Life.

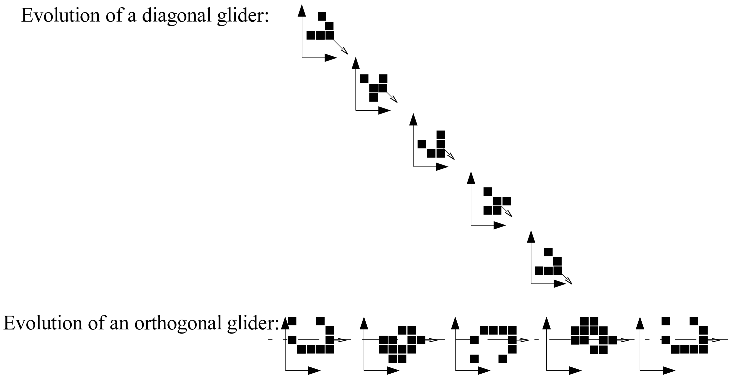

Evolution of a diagonal glider:
Evolution of an orthogonal glider:

Figure 9. Evolutions of orthogonal and diagonal gliders for generations from 0 to 5 of the Game of Life. For the diagonal glider $T = 4$, $D_x = -1$ and $D_y = -1$ and for the orthogonal glider $T = 4$, $D_x = 0$ and $D_y = 1$.

2.7. Glider Gun

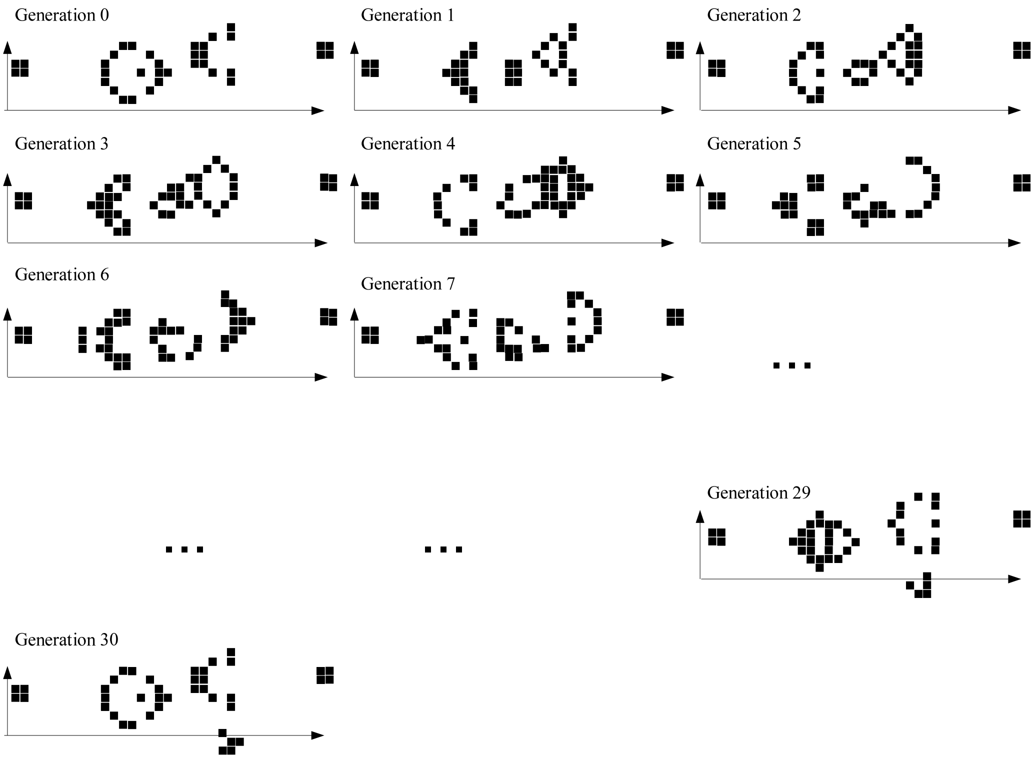
A pattern G is a *glider gun* of the automaton A if and only if the evolution of G in the automaton A is such that there exists a non-zero natural number t such that for all natural numbers n, we get:

- For all integers x and y, $G^A_{(n+1)\times t}(x,y) = 1$ if $G^A_{n\times t}(x,y) = 1$.
- Let P be a pattern for which for all integers x and y we get:
$$P_0^A(x,y) = G^A_{(n+1)\times t}(x,y) - G^A_{n\times t}(x,y).$$

  The pattern P is such that:

- there exists two integers x and y such that $P_0^A(x,y) = 1$.
- For all integers x and y such that $P_0^A(x,y) = 1$ we get (x,y) is in a glider included in P.

The number t is called the period of the glider gun. The gliders included in the pattern P is said emitted by G. Figure 10 shows the gosper gun of the Game of Life.



Generation 0

Generation 1

Generation 2

Generation 3

Generation 4

Generation 5

Generation 6

Generation 7

Generation 29

Generation 30

Figure 10. Evolutions of the Gosper gun of the Game of Life.

The period of Gosper Gun is 30 so for all integers x and y, $G_{30}^{Life}(x,y) = 1$ if $G_0^{Life}(x,y) = 1$. Figure 11 shows the pattern P, in which for all integers x and y, we get:

$$P_0^{Life}(x,y) = GosperGun_{30}^{Life}(x,y) - GosperGun_0^{Life}(x,y).$$

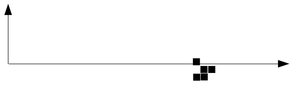


Figure 11. The pattern P_0^{Life}.

All the living cells of P_0^{Life} are included in a glider of Game of Life.

3. Game of Life

The Game of Life, discovered by Conway in 1970 and popularised by Gardner in [2], is the most well known universal automaton. The transition rule of the Game of Life will be described and then the simulation of the AND and NOT gates of the Game of Life will be breifly exposed.

3.1. Transition Rule

The Game of Life is an automaton of I ($\mathbb{Z}^2$,$\{0,1\}$,Moore neighbourhood,$\delta : S^9 \mapsto S$). The state of a cell of the Game of Life at the next generation depends on its own state and the sum of cells in state 1 among its eight direct neighbours.

The behaviour of the cells of the Game of Life is inspired by the ones of real cells as a cell could die if there not enough cells surrounding it or if there too many. So if a cell in state 1 has 0 or 1 neighbours in state 1 then this cell will be in state 0 at the next generation. If a cell in state 1 has more than 3 neighbours in state 1 then the state of this cell will be 0 at the next generation. A birth happens if a cell has three neighbours in state 0 so if a cell in state 0 has 3 neighbours in state 1 then the state of this cell will be 1 at the next generation.

These rules can be formulized saying that the Game of Life is the automaton of I ($\mathbb{Z}^2$,$\{0,1\}$,Moore neighbourhood,$\delta : S^9 \mapsto S$) such that for all binary numbers $x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8$ and x_9 we get:

$$\delta\begin{pmatrix} x_1,x_2,x_3, \\ x_4,x_5,x_6, \\ x_7,x_8,x_9 \end{pmatrix} = \begin{cases} 1 & \text{if} \quad x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8+x_9=3 \\ 1 & \text{if} \quad x_1+x_2+x_3+x_4+x_6+x_7+x_8+x_9=2 \\ 0 & \text{otherwise} \end{cases}$$

3.2. AND Gate

Conway et al. [6] have shown the universality of the Game of Life using logic gates. In this simulation, Conway used streams of gliders as voltage to carry information. A stream emitted by a gun is a series of gliders periodically spaced whereas in a stream carrying information a glider may exist or not depending on the logic value 0 or 1 that is presented.

Collisions between different streams are used to realize an AND gate and then a NOT gate.

The AND gate is based on a special collision between two streams A and B of gliders, called in [6] vanishing reaction and shown in figure 12, which has the following result:

- If a glider is present in both streams, the two gliders annihilate each other completely before the next gliders arrive.
- If a glider is present in stream A or B but not in the other stream, the glider which is present continue its run.
- If no glider is present in streams A and B nothing happens.

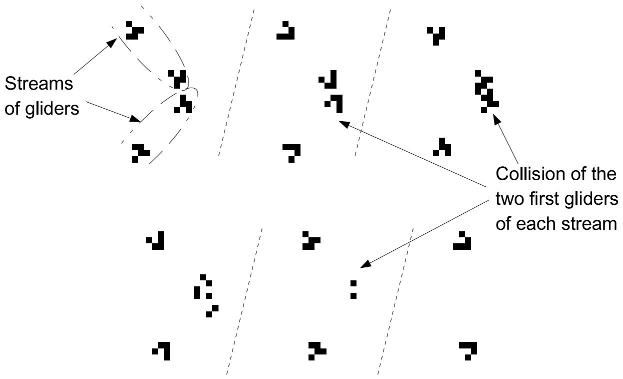


Figure 12. Collision between two streams of gliders.

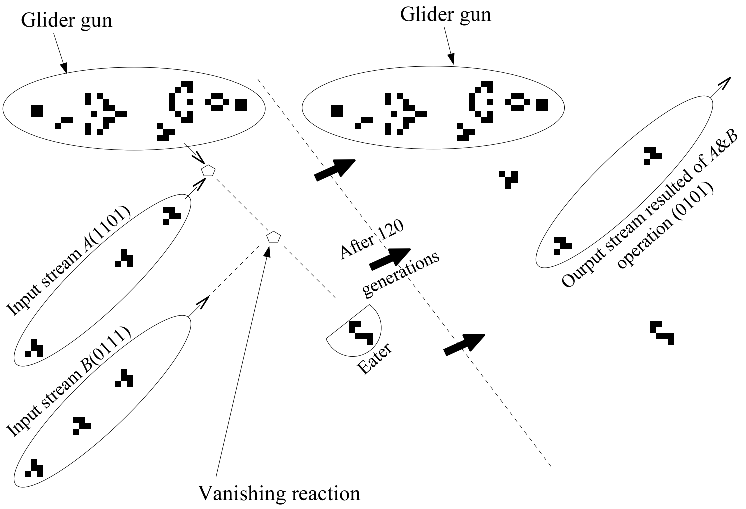
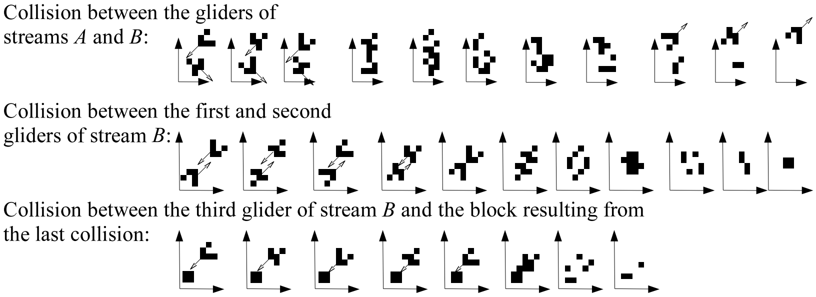


Figure 13. AND gate simulated by the Game of Life [6].

An AND gate is simulated in figure 13 with two input streams A and B. The glider gun used by this simulation is the gosper gun [6] and emits a new glider every 30 generations. This gun creates a glider stream that ”crashes” stream A. If a glider is present in stream A, the two gliders are destroyed by this collision, otherwise the glider emitted by the gun continues its run. The stream resulting from this first collision, at right angle to stream A, is $\overline{A}$. This stream crashes stream B producing a stream aligned with the stream B, which is the result of the operation *A and B*. The synchronization and the position of the different components are critical to the proper function of the simulation.

3.3. NOT Gate

The NOT Gate is based on a collision between a glider g and a stream A emitted by a gun, called in [6] kickback reaction, which destroys the glider of the stream A and turns back the glider g. Conway shown that this collision allows to build guns emitting streams with gliders spaced by any multiple of 4 of how many gliders are spaced in a stream emitted by



Collision between the gliders of streams *A* and *B*:

Collision between the first and second gliders of stream *B*:

Collision between the third glider of stream *B* and the block resulting from the last collision:

Figure 14. Kickback reaction between two streams step by step.

the gosper gun. Streams emitted by this kind of guns and the kickback reaction is used in the simulation of a NOT gate.

The kickback reaction between a stream A and a stream B emitted by a gun has the following result:

- If a glider is present in stream A, this glider is destroyed and the glider of stream B turns back in the opposite direction as shown figure 14. This glider collides with the second glider of stream B and they destroyed each other creating a block of four stable cells as shown figure 14. The third glider of stream B collides with this block and they destroy each other. Finally, these collisions have destroyed three gliders of the stream B.

- If a glider is not present in stream A nothing happens and gliders of stream B continue their run.

In the simulation of a NOT gate as shown figure 15, the input stream is a spaced stream in which each bit is followed by a glider. This stream collides with a stream of gliders emitted by a gun. The result of this collision depends on the value of the first bit A_1 of the stream A:

- If A_1 equals 1, this glider collides with a glider emitted by the gosper gun and is destroyed. The glider emitted by the gun is kickback and destroys the two next gliders emitted by the gun. So the fourth glider emitted by the gun survives. Then, the stream emitted by the gosper gun represents the values 1000.

- If A_1 equals 0, the glider emitted by the gosper gun continue its run. The glider following A_1 collides with the second glider emitted by the gun which is kickback and destroys the two next gliders emitted by the gun which are the third and fourth gliders emitted by the gosper gun. Then the stream emitted by the gosper gun represents the values 0001.

As the stream emitted by the gosper gun represents $A_1 00\overline{A_1}$, a vanishing collision with a spaced stream is realized to obtain an output stream lined up with the input stream and containing $\overline{A_1}$.

In order to simulate a NAND gate with a cellular automaton with the method of demonstration of the game of life, the given automaton needs to accept

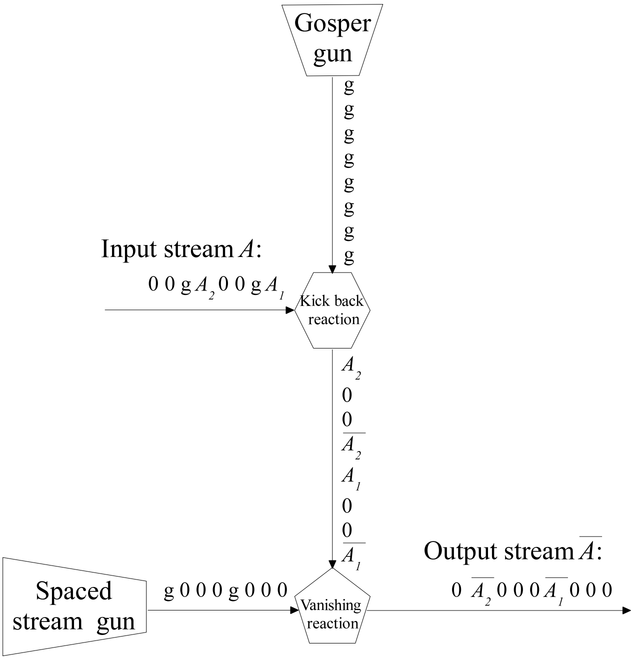

Gosper gun
g g g g g g g g
Input stream A:
$0\ 0\ g\ A_2\ 0\ 0\ g\ A_1$
Kick back reaction
A_2 0 0 $\overline{A_2}$ A_1 0 0 $\overline{A_1}$
Output stream $\overline{A}$:
$0\ \overline{A_2}\ 0\ 0\ 0\ \overline{A_1}\ 0\ 0\ 0$
Spaced stream gun
g 0 0 0 g 0 0 0
Vanishing reaction

Figure 15. NOT gate simulated by the Game of Life [6].

- a glider gun,
- a vanishing reaction,
- the three steps of a kickback reaction.

4. Gliders

In order to search for new automata accepting gliders evolutionary computing is used. An evolutionary algorithm [14] incorporates aspects of natural selection or survival of the fittest. It maintains a population of structures (usually randomly generated to begin with) that evolve according to rules of selection, recombination, mutation, and survival referred to as genetic operators. A shared ”environment” determines the fitness or performance of each individual in the population. The fittest individuals are more likely to be selected for reproduction (retention or duplication), while recombination and mutation modify those individuals, yielding potentially superior ones.

The first subsection describes the evolutionary algorithm that found the gliders described in the second subsection.

4.1. Evolutionary Algorithm

Search Space An evolutionary algorithm is used in order to find automata accepting gliders in the space I described in Section 2. An automaton of this space can be described

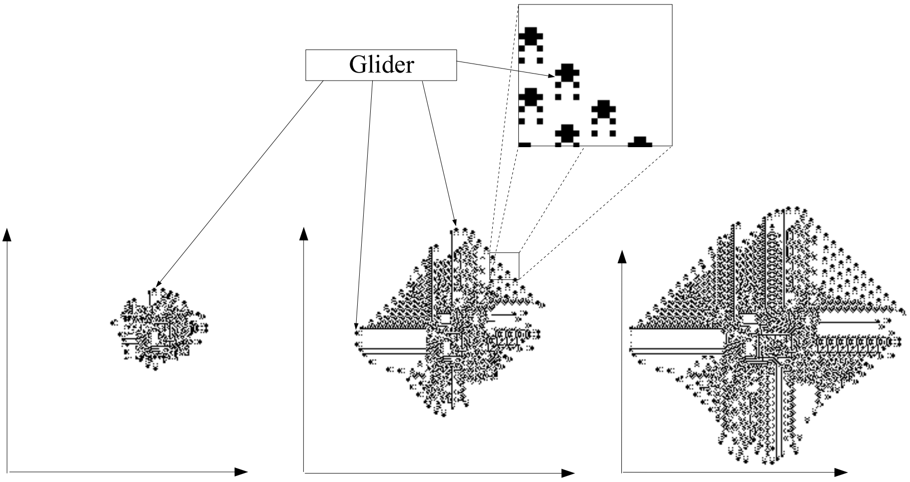


Figure 16. Result of the evolution of a random configuration of cells by an automata discovered by an evolutionary algorithm.

by telling what will become of a cell in the next generation, depending on its neighbours. An individual is an automaton coded as a bitstring of 102 booleans (cf. figure 7) representing the value of a cell at the next generation for each neighbourhood.

Fitness Function Several fitness function were tried for our algorithm. A first function was used but its results were not the expected results so this function was modified in order to obtain the final fitness function.

The first fitness function attempts to maximise the number of gliders that appear during the evolution of a random configuration of cells by the tested automaton. The discovered automata accepted gliders but a random configuration of cell evolving by these rules grows infinitely. Figure 16 shows a typical evolution of a random configuration of cells by a discovered rule. A second fitness function is tried.

So the evolutionary algorithm attempts to maximise the number of gliders × the number of periodic patterns that appear during the evolution of a random configuration of cells by the tested automaton. (A more detailed description of the gliders and periodic patterns detector (inspired by Bays [7]) can be found in [8].)

Initialisation The 102 bits of each individual are initialised at random.

Genetic Operators The mutation function simply consists of mutating one bit among 102, while the recombination is a single point crossover with a locus situated exactly on the middle of the genotype. This locus was chosen since the first 51 neighbourhoods determine the birth of cells, while the other 51 determine how they survive or die.

Evolution Engine It is very close to a $(\mu + \lambda)$ Evolution Strategy [14], although on a bitstring individual, and therefore without adaptive mutation : the population is made of 20 parents that are selected randomly to create 20 children by mutation only and 10

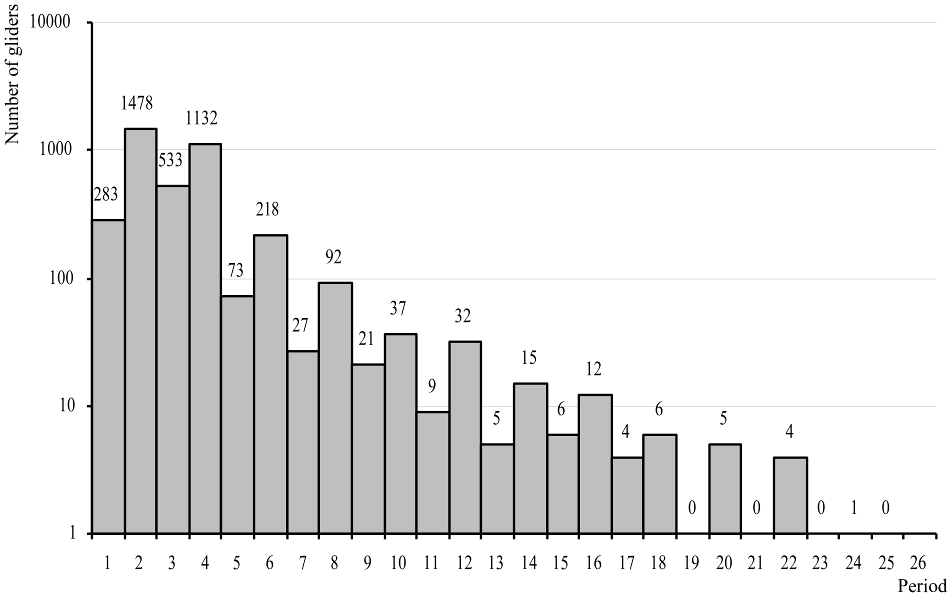


Figure 17. Distribution of the period of the discovered orthogonal gliders.

others by recombination. As in a straight ES+, the 20 best individuals among the 20 parents + 30 children are selected to create the next generation.

Stopping Criterion The algorithm stops after 200 generations, which, on a 800Mhz PC Athlon, takes around 20 minutes to complete.

4.2. Result

This study takes into account the gliders found after one hundred hours of execution of our algorithm on a 800Mhz PC Athlon. 4660 Gliders of different movements, velocities, periods and directions were found. No oblique glider was found. Orthogonal gliders are studied first then diagonal gliders are studied.

4.2.1. Orthogonal Gliders

3993 orthogonal gliders were discovered. The distribution of the period of these gliders is shown figure 17.

Each glider has a movement of a given number of cells during its period. The lower glider period is 1. Gliders of period 1, as shown in figure 18 for few of them, have a movement of one cell per period. The most common period is 2 and some gliders of period 2 with a movement of 1 and 2 are shown figures 19 and 20. Gliders of period 3 are less common than gliders of period 2 and figures 21 and 22 show gliders of period 3 moving one and three cells during their period. The odd periods are the most common. That can be explained because some of the gliders that were discovered recover a symmetry of their original shape after half of their period has ellapsed as shown figure 23 for a glider of period 4 and movement 4.

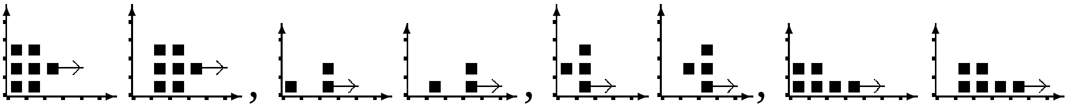


Figure 18. Gliders of period 1 at generations 0 and 1.

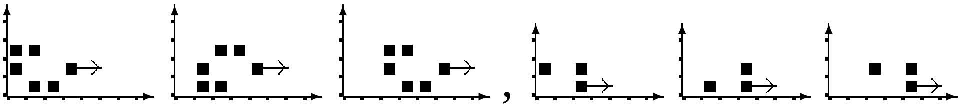


Figure 19. Gliders of period 2 and a movement of 2 at generations 0, 1 and 2.

The tabular 24 shows for each period and movement during a period, the number of gliders that were discovered.

The movement during a period has to be at least 1 considering the definition of a glider, and can not be higher than the period because it is impossible for a glider to have a movement higher than 1 cell per generation, this is called the *speed of light* [7].

Tabular 24 shows three main categories of orthogonal gliders:

First Category Gliders of the first category have a period higher than one and have a movement of one cell during their period. It is the first column on tabular 24. The number of discovered gliders of this category deacreses gradually when the period increases.

Second Category Gliders of the second Category have a movement of n cells during a period of n. It is the diagonal on tabular 24. The number of discovered gliders of this type with odd periods is higher than the number of discovered gliders of this category with even periods and deacreses when the period increases.

Third Category Gliders of the third category have a movement of $\frac{n}{2}$ cell during a period of n. Only gliders with odd periods can be of this category. The number of discovered gliders of this category decreases when the period increases.

The distribution of the velocity of orthogonal gliders is shown figure 25.

The velocity of a glider is its movement divided by its period, thus an integer divided by its period. Therefore for a glider of period N, the velocity can only be $\frac{D}{N}$ where D is an integer between 1 and N.

47.5 percent of the discovered gliders have a velocity of 1 that is the highest velocity.

4.2.2. Diagonal Gliders

668 diagonal gliders were discovered. The distribution of the period of these gliders is shown in figure 26.

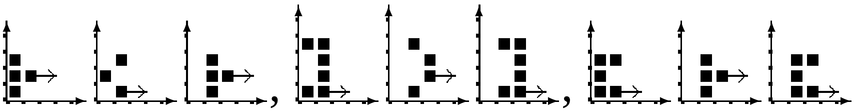


Figure 20. Gliders of period 2 and velocity $\frac{1}{2}$ at generations 0, 1 and 2.

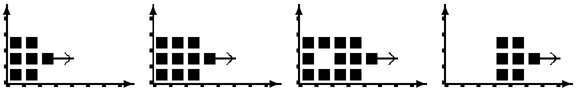


Figure 21. A glider of period 3 and a movement of 3 at generations 0, 1, 2 and 3.

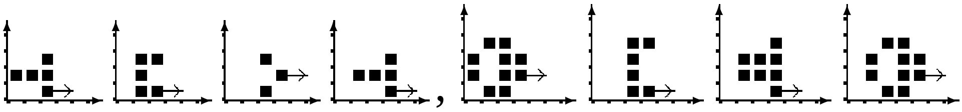


Figure 22. Gliders of period 3 and a movement of 1 at generations 0, 1, 2 and 3.

Each glider has a movement of a given number of cells during its period. The lower glider period is 2 and the two gliders of this period are shown in figure 27. There are 102 gliders of period 3 that are discovered and a sample of them is shown figure 28. The most common period is 4 and some gliders of period 4 with a movement of 1 are shown figure 29. As for orthogonal gliders, the odd periods are the most common and that can be explained with the same reasons. The number of discovered gliders decreases regularly when the period increases but there are less gliders of period 9 than gliders of period 11 and more gliders of period 14 than gliders of period 12.

The tabular 30 shows for each period, and movement during a period, the number of gliders that were discovered.

The movement during a period has to be at least 1 and, considering the definition of a diagonal glider, it can not be higher than the half of the period. The movement of discovered diagonal gliders is not as high as the movement of the discovered orthogonal gliders as only a glider of period 16 have a movement higher than 4 and only two gliders has a movement of 4.

The distribution of the velocity of these gliders is shown figure 25.

The velocity of a glider is its movement divided by its period, so an integer divided by its period. So for a glider of period N, the velocity can only be $\frac{D}{N}$ where D is an integer between 1 and N.

5. Universality

The algorithm described above provided several automata accepting gliders. Among the discovered automata, some would surprisingly generate glider guns for nearly every evolution of a random cell configuration. The first of them discovered, R_0, described in the first subsection, is picked up as a potential candidate for a universal automaton.

The demonstration of its universality is inspired by the demonstration of universality of the Game of Life. In the demonstration of universality of the Game of Life an AND gate is

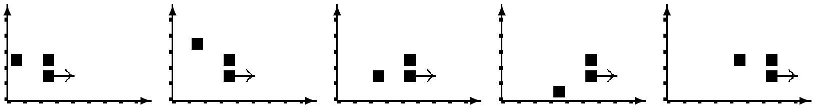


Figure 23. A glider of period 4 at generations 0, 1, 2, 3 and 4.

		Moving													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
	1	283													
	2	404	1074												
	3	308	0	225											
	4	255	359	0	518										
	5	61	12	0	0	0									
	6	25	62	42	0	0	89								
	7	18	4	5	0	0	0	0							
	8	15	47	4	14	0	0	0	12						
	9	11	5	2	0	0	0	0	0	3					
	10	3	20	3	4	6	0	0	0	0	1				
	11	3	4	1	1	0	0	0	0	0	0	0			
P	12	1	11	2	1	0	9	0	0	0	0	0	8		
e	13	3	2	0	0	0	0	0	0	0	0	0	0	0	
r	14	1	7	2	1	1	0	2	0	0	0	0	0	0	1
i	15	2	2	1	1	0	0	0	0	0	0	0	0	0	0
o	16	0	9	0	1	0	1	0	1	0	0	0	0	0	0
d	17	2	1	0	1	0	0	0	0	0	0	0	0	0	0
	18	1	5	0	0	0	0	0	0	0	0	0	0	0	0
	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	20	0	2	1	1	0	0	0	0	0	1	0	0	0	0
	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	22	1	2	0	1	0	0	0	0	0	0	0	0	0	0
	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	24	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24. Numbers of orthogonal gliders that were discovered for each period and movement. A blank cell means gliders of this combination can not exist.

simulated. To simulate an AND gate with the same method, one needs an eater in R_0. The search for eater is described in the first subsection and an eater was found in an automaton R very close to R_0. The second step is to find a configuration of glider guns and eaters that can simulate a NAND Gate. In the Game of Life, it was done thanks to the kickback reaction. The kickback reaction was searched for in R without being found. Another configuration of gliders and glider guns, described in the second subsection, in R simulated an NAND gate. The third subsection describes how to use this simulation of a *NAND* Gate to simulate one cell of the Game of Life. The final step described in the last subsection is to tile the plan of the Game of Life by identical simulation of cells in order to prove the universality of R. All these steps are described in the following subsection.

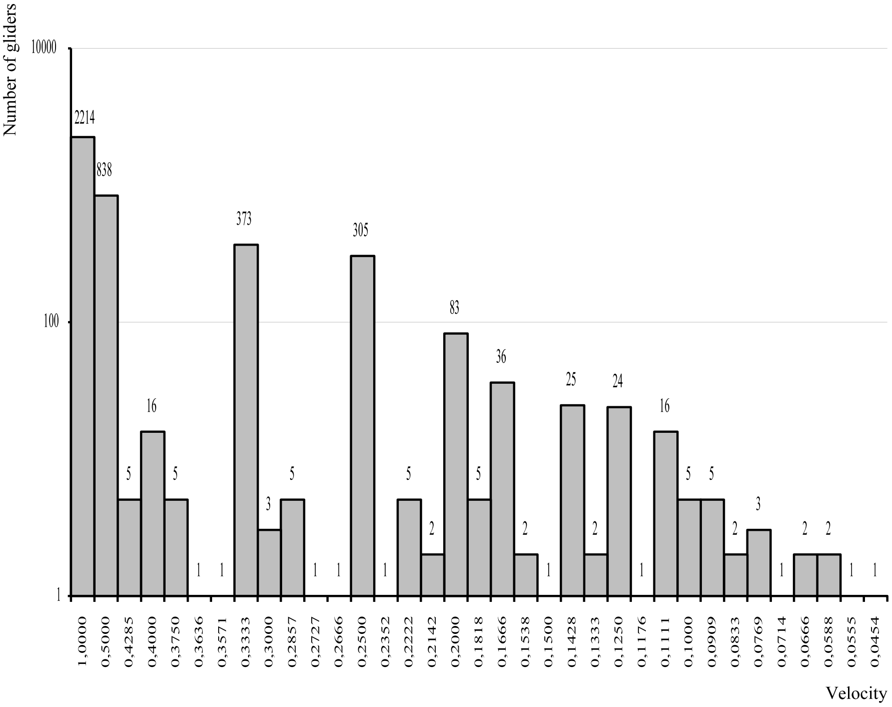


Figure 25. Distribution of the velocity of the discovered orthogonal gliders.

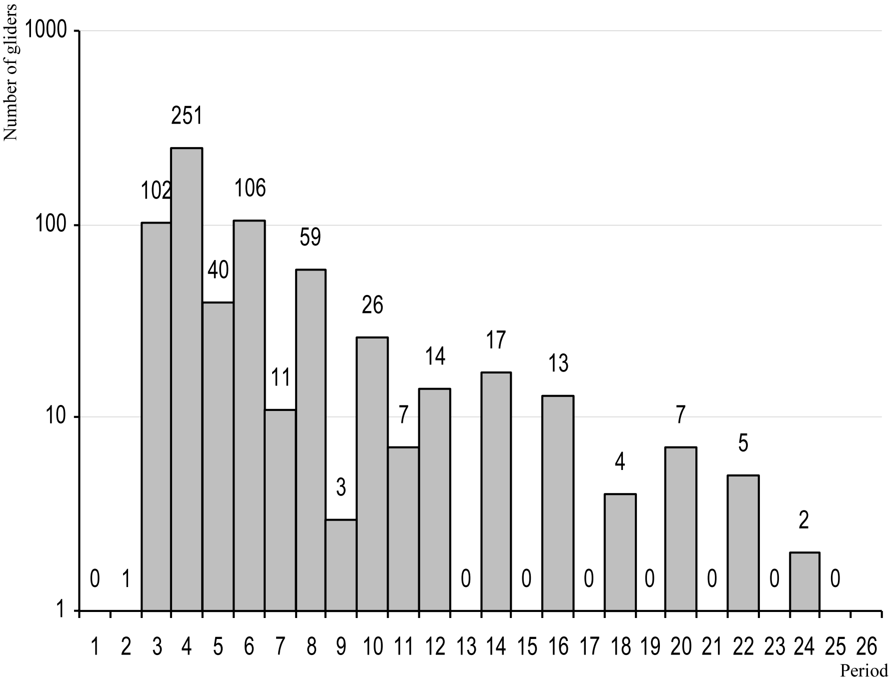


Figure 26. Distribution of the period of the discovered diagonal gliders.

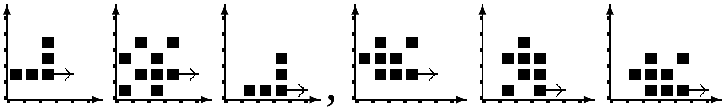


Figure 27. Gliders of period 2 at generations 0, 1 and 2.

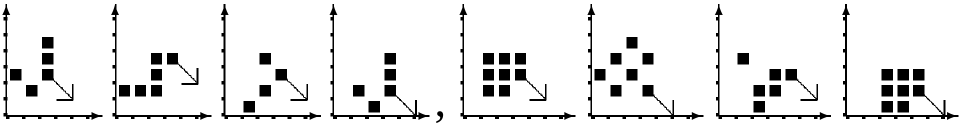


Figure 28. Gliders of period 3 at generations 0, 1, 2 and 3.

5.1. The R_0 Automaton: an Experimental Result

The 102 different neighbourhoods of R_0 can be visually presented as in figure 32, and figure 33 shows the glider gun G_0 that appeared spontaneously from a random configuration of cells.

5.2. Looking for an "Eater"

The automaton R_0 that was discovered in the previous section accepts gliders and a glider gun G_0. The gun G_0 fires one glider toward each cardinal point whereas the Game of Life one fires just one glider. In order to use the glider gun G_0 like the Game of Life, three glider streams must be removed. An eater can be used to suppressed stream (cf. figure 34) so an eater is searched for.

Eaters are periodic patterns that, after the absorption of a glider, resume their original shape and position quickly enough to absorb another arriving glider (cf. figure 34).

5.2.1. Evolutionary Algorithm

An eater was manualy search for in R_0. As no eater was found in R_0, the search space of automata became all automata accepting the glider gun G_0. So an automaton R is searched for in this space that accepts both the glider gun G_0 and an eater. This space was determined by deterministically finding which of the 102 neighbourhoods of automaton R_0 were needed for gun G_0 to operate normally. It turns out that G_0 uses 81 different neigbourhoods meaning that the output of the 21 other neighbourhoods, shown in figure 37, could be changed. This, our search space of automata contain 2^{21} elements.

An eater being a periodic pattern, a collection of 10 small periodic patterns of R_0 appearing frequently *and using only* neighbourhoods among the 81 ones necessary for G_0 were chosen. figure 35 shows this collection. Those periodic patterns were therefore sure to appear in all of the 2^{21} automata implementing G_0.

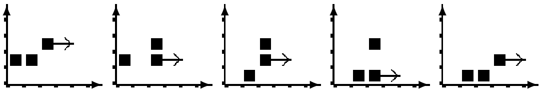


Figure 29. Gliders of period 4 at generations 0, 1, 2, 3 and 4.

		Moving											
		1	2	3	4	5	6	7	8	9	10	11	12
	1												
	2	1											
	3	102											
	4	251	0										
	5	40	0										
	6	88	18	0									
	7	10	1	0									
	8	45	14	0	0								
	9	3	0	0	0								
	10	21	5	0	0	0							
	11	4	1	2	0	0							
P	12	7	2	4	1	0	0						
e	13	0	0	0	0	0	0						
r	14	9	7	1	0	0	0	0					
i	15	0	0	0	0	0	0	0					
o	16	5	8	0	0	0	0	0	1				
d	17	0	0	0	0	0	0	0	0				
	18	3	0	1	0	0	0	0	0	0			
	19	0	0	0	0	0	0	0	0	0			
	20	1	4	1	1	0	0	0	0	0	0		
	21	0	0	0	0	0	0	0	0	0	0		
	22	3	2	0	0	0	0	0	0	0	0	0	
	23	0	0	0	0	0	0	0	0	0	0	0	
	24	1	1	0	0	0	0	0	0	0	0	0	0

Figure 30. Numbers of diagonal gliders that were discovered for each period and movement. A blank cell means gliders of this combination can not exist.

Finally, in order to find an eater, one needs to perform on the established collection of periodic patterns what could be called a crash test: each periodic pattern is positioned in front of a stream of gliders, and its fitness is simply the number of crashes it survives.

The number of possibilities being quite large (10 patterns to be tested in different relative positions with reference to the stream of gliders among 2^{21} different automata), a second evolutionary algorithm was therefore created, with the following characteristics:

Individual Structure An individual is made of:

- a 21 bit bitstring determining one automaton among 2^{21} possible ones,
- an integer between 1 and 10, describing one pattern among the 10 chosen periodic patterns,
- the relative position of the pattern relatively to the stream of gliders, coded by two integers, x and y, varying between $[-8,8]$ and $[0,1]$.

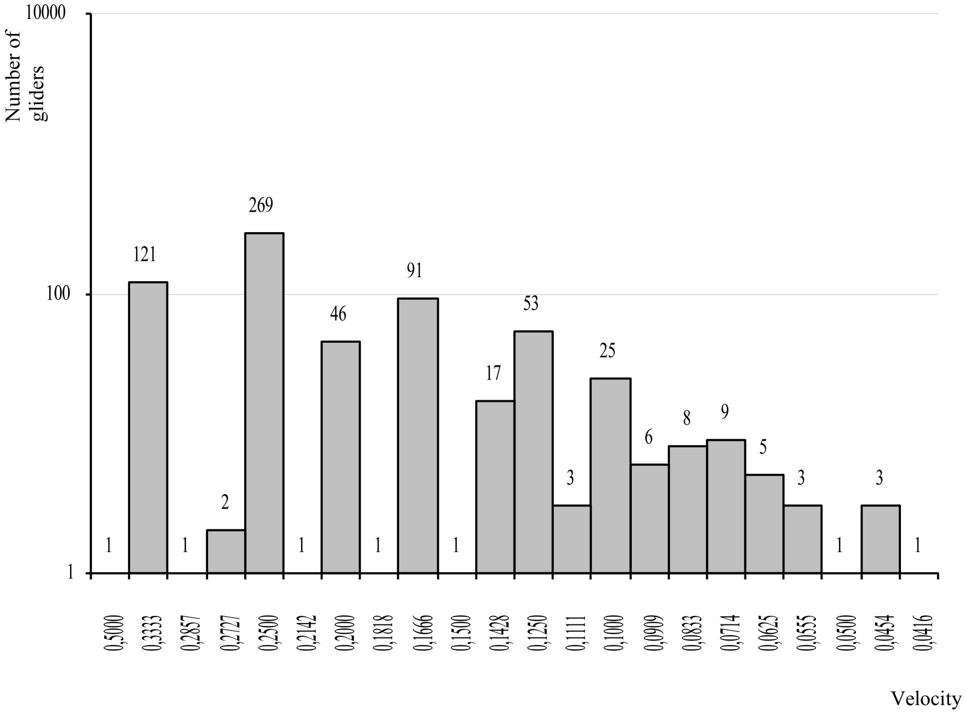
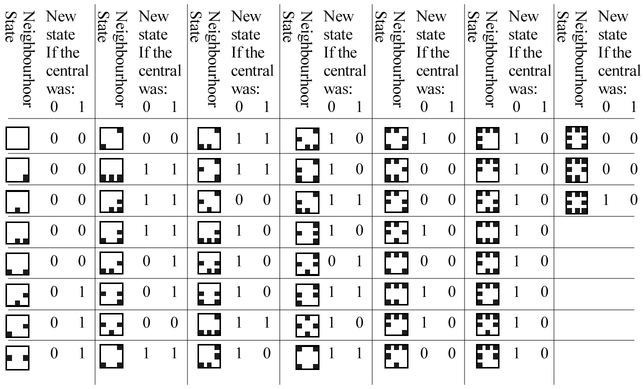


Figure 31. Distribution of the velocity of the discovered diagonal gliders.



Neighbourhoor State	New state If the central was:		Neighbourhoor State	New state If the central was:		Neighbourhoor State	New state If the central was:		Neighbourhoor State	New state If the central was:		Neighbourhoor State	New state If the central was:		Neighbourhoor State	New state If the central was:		Neighbourhoor State	New state If the central was:	
	0	1		0	1		0	1		0	1		0	1		0	1		0	1
	0	0		0	0		1	1		1	0		1	0		1	0		0	0
	0	0		1	1		1	1		1	0		0	0		1	0		0	0
	0	0		1	1		0	0		1	1		0	0		1	0		1	0
	0	0		1	1		1	0		1	0		1	0		1	0			
	0	0		0	1		1	0		0	1		0	0		1	0			
	0	1		0	1		1	0		1	1		1	0		1	0			
	0	1		0	0		1	1		1	0		1	0		1	0			
	0	1		1	1		1	0		1	1		0	0		1	0			

Figure 32. The transition rule of the cellular automaton R_0.

Individuals are initialised with R_0, a random integer between 1 and 10, and randomly within their interval for x and y.

Fitness Function Number of gliders stopped by an individual.

Genetic Operators The only operator is a mutator, since no really “intelligent” recombination function could be elaborated. The mutator is therefore called on all created offsprings and can either:

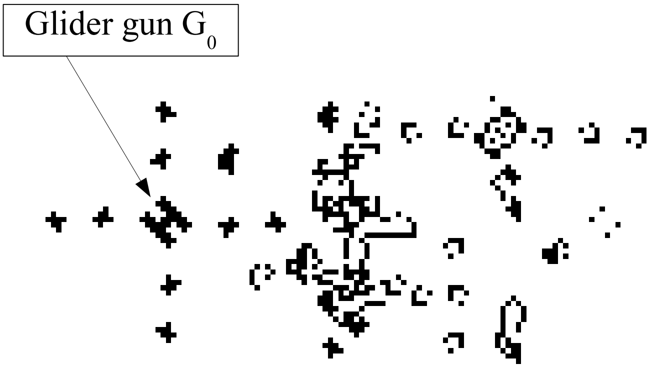


Figure 33. An evolution of a random configuration of cell by R_0 showing G_0.

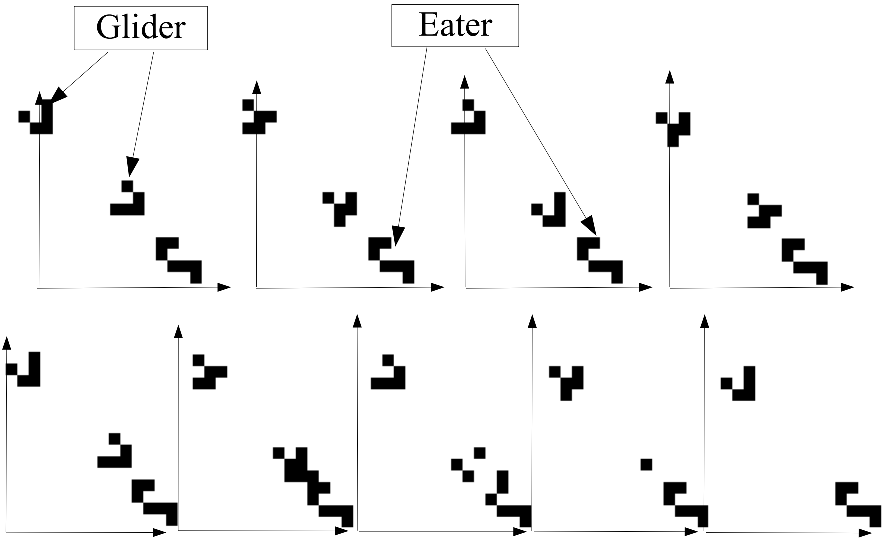


Figure 34. Eater in the Game of Life.

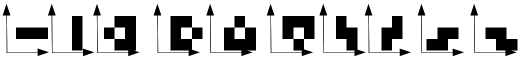


Figure 35. 10 small periodic patterns of R_0 appearing frequently.

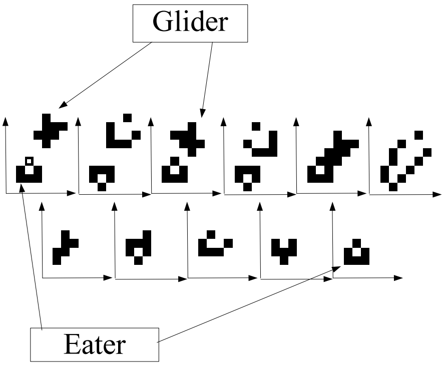


Figure 36. Eater of R in action, found by a second evolutionary algorithm.

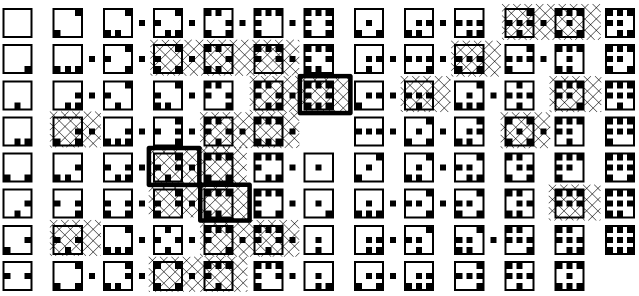


Figure 37. A black cell on the right of the neighbourhood indicates a future central cell. The R automaton can then be represented by 0000000001 1100011101 1111111111 1110010010 1111111100 0000001110 1111101110 0001000101 1010000000 0000000000 00. The neighbourhoods which are in a box are the ones for which the corresponding value is different for R_0 and the neighbourhoods in the hatching aera is not used by G_0.

- choose any pattern among the 10 available,
- mutate one bit in the bitstring,
- move the position of the pattern by ±1 within the defined boundaries for x and y.

Evolution Engine It is this time closer to an Evolutionary Programming Engine, since it has no crossover, although the EP tournament was not implemented. 25 children are created by mutation of 25 parents. Among the 50 resulting individuals, the 25 best are selected to create the next generation.

Stopping Criterion Discovery of an eater that would survive 50 000 collisions.

5.2.2. The Eater of the *R* Automaton : an Experimental Result

This algorithm allowed to discover the automaton R accepting both the glider gun G_0 and an eater shown figure 36.

Figure 37 shows that the automaton R is different from R_0 by just by three bits.

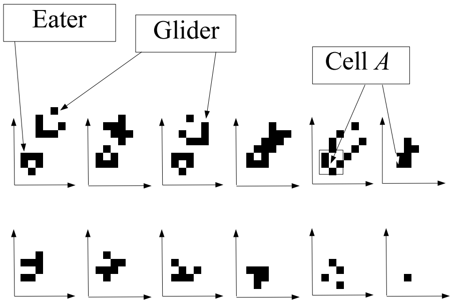


Figure 38. Eater of R evolving by R_0.

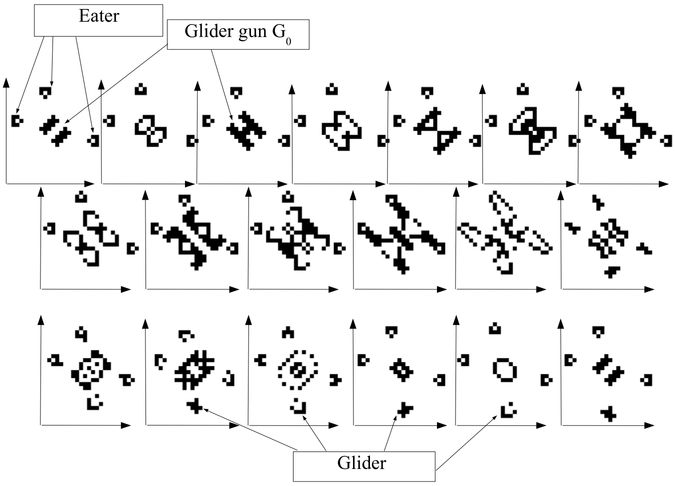


Figure 39. The glider gun G generation by generation.

The role of the modified bit between R and R_0 is shown figure 38 as the discovered eater evolving by the rule R_0. In this figure, the first cell with one of the three neighboorhoods that change between R and R_0 is called A. This cell becomes state 1 with R_0 at the next generation whereas it becomes state 0 with R. So the evolution of the eater is different in R_0.

The eater is used to suppress the three extraneous streams of gliders produced by the glider gun G_0 in order to build the gun G shown figure 39.

5.3. NAND Gate

In this simulation, binary numbers are implemented as a finite streams of gliders, where gliders represent 1s and missing gliders represent 0s. This simulation is based on collisions between streams of gliders. These collisions will be listed then they will be used to built new patterns then how to build a *NAND* Gate from these patterns will be described.

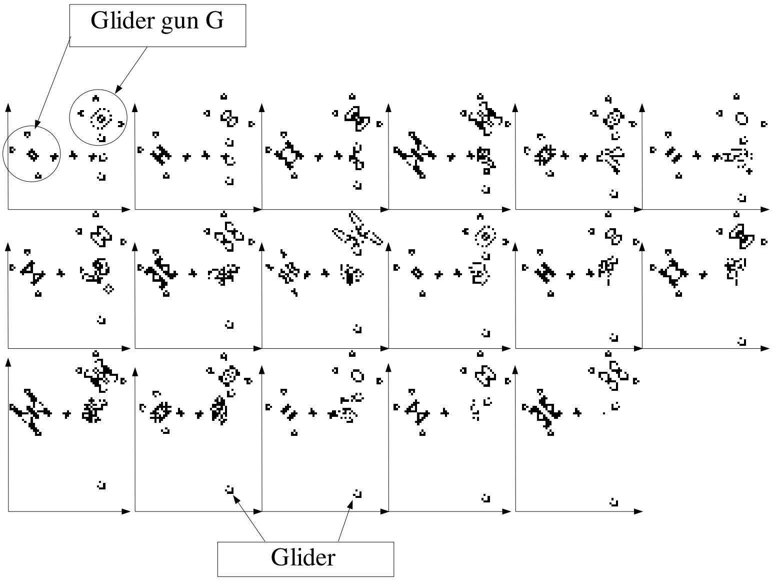


Figure 40. A 5-thining of *R*, viewed every 4 generations.

5.3.1. Collisions

Several collisions between streams of gliders are described in this section. Four collisions - namely thining collision, process collision, vanishing collision, and duplicative collision - are at right angles whereas the last one is a frontal collision:

Thining Collision A thinning collisions is a collision of two streams of gliders which destroy one of the streams but let a glider survive every periodic number of generations in the other one. A collision is n-thinning if one glider per *n* survives. Figure 40 shows a 5-thinning collision between two streams emited by two guns. So in the survived stream gliders are spaced by 45 cells.

Process Collision A process collision is a collision of two perpendicular streams of gliders which destroys one of the streams but lets the other one survive. After the collision, the gliders of the surviving stream are of another kind, called *large gliders* as shown in Figure 41. The front cells of the large gliders are identical to those of the standard gliders. Therefore, the speed and spacing of the large gliders is identical to the speed and spacing of the original standard gliders. Large gliders can be seen as being fat standard gliders.

Vanishing Collision A vanishing collision is a collision of two streams of gliders or large gliders which destroy the two streams. If there is a missing glider or a missing large glider in one stream then the corresponding glider in the other stream survives. The result is the following for the two streams:

- If there is a glider in each stream, after the collision there is a missing glider in each stream

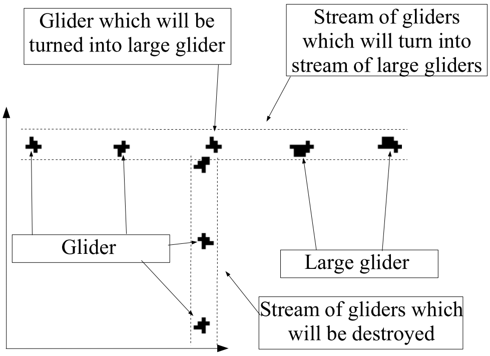

Glider which will be turned into large glider
Stream of gliders which will turn into stream of large gliders
Glider
Large glider
Stream of gliders which will be destroyed

Figure 41. Process collision between two gliders.

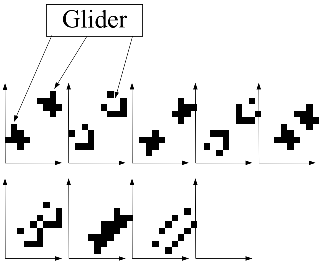


Figure 42. Vanishing collision between two gliders.

- If there is a missing glider in the one of the two streams, after the collision the corresponding glider survives in the other stream
- If there is a missing glider in each stream, after the collision there is a missing glider in each stream

So the stream A becomes the result of the operation A *and* $\overline{B}$ and the stream B becomes B *and* $\overline{A}$. If the stream A is a full stream then it becomes $\overline{B}$ whereas the stream B is destroyed.

Frontal Collision A frontal collision is a collision of two identical streams of large gliders which produces two orthogonal standard streams of gliders equal to the input streams as shown figure 43.

Duplicative Collision A duplicative collision is a collision of two streams of gliders which destroys one stream, called A, but lets the other one, called B, survive as shown Fig 44. If there is a missing glider in stream B then the corresponding glider in stream A survives. So stream B is unchanged by this collision and the result is the following for stream A:

- If there is a missing glider in stream A, after the collision there is a missing glider in stream A
- If there are a glider in stream A and a glider in stream B, after the collision there is a missing glider in stream A

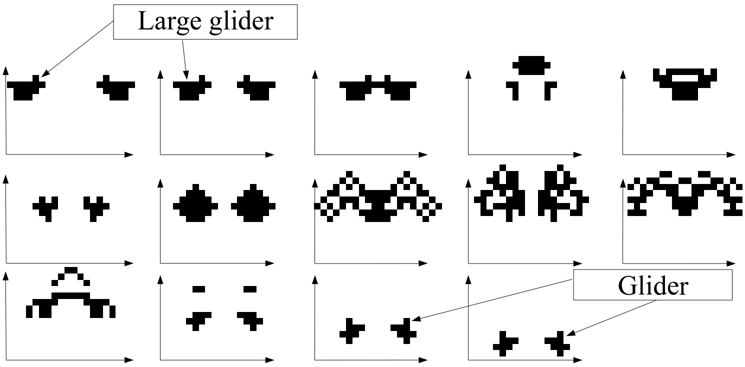


Figure 43. Frontal collision between two large gliders.

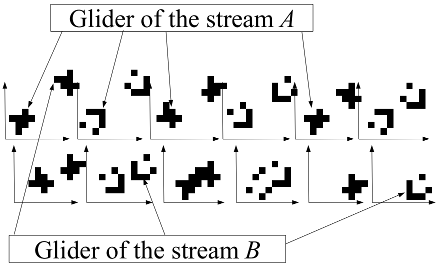


Figure 44. Duplicative collision between two gliders.

- If there are a glider in stream *A* and a missing glider in stream *B*, after the collision there is a glider in stream *A*

So the streams *A* become the result of the operation *A and* $\overline{B}$. If the stream *A* is a full stream then it becomes $\overline{B}$ whereas the stream *B* survives.

5.3.2. New Pattern

Unfortunately, the cellular automaton implementing a cell of the Game of Life would be too difficult to explain by showing groups of cells on a grid, let alone a CA implementing a cell of the Game of Life. Therefore, a much clearer analytical description was needed, that should also allow replicability of the contents of this paper.

In order to simplify the representation of a CA, one can replace its building blocks by an analytical description, made of a letter referring to the pattern followed by three parameters (D,x,y) where *D* denotes a direction (*N*orth, *E*ast, *S*outh, *W*est) and *x*, *y* the coordinates of a specific cell of the pattern (cf. [9]). An arrow is added in graphic descriptions to help visualising the CA. The analytical description of a glider stream and an eater are given so the complex glider gun and the large glider gun are built.

Glider Stream Figure 45 shows a glider stream $S(E,x,y)$, where *x* and *y* are the coordinates of the white cell whence an arrow is shooting.

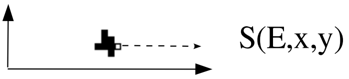


Figure 45. A glider stream and its analytical representation $S(E,x,y)$.

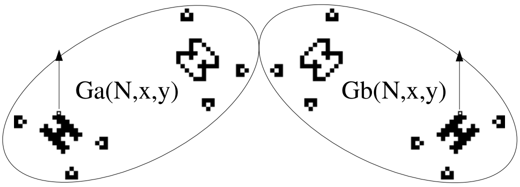


Figure 46. Complex gun of R, viewed at generation 2. x,y are the coordinates of the white cell, whence an arrow is shooting.

Eater The eater of figure 36 can be identified as $E(N,x,y)$, where N denotes its northward orientation and x,y denote the position of the white cell.

Complex Glider Gun The glider gun of figure 33 shoots gliders spaced only every nine cells. Unfortunately the duplicative collision needs gliders that are more closely spaced. In order to build a gun shooting such gliders, the thinning collision shown in Fig 40 is used. So the complex gun, shown in figure 46, shoots gliders spaced by 45 cells, which gives more slack to work on streams. Figure 46 shows instances of this gun, used later on in this paper, namely $Ga(S,x,y)$ and $Gb(S,x,y)$. The complex guns in other cardinal directions are obtained by rotation.

Large Glider Gun Thanks to a process collision, a large gun (L) shooting a large glider every 45 cells is made of two complex guns G shooting their stream perpendicularly (cf. figure 47).

5.3.3. Assembling Patterns into a *NOT* Gate

On figure 48, a *NAND* gate is built thanks to the patterns of the previous section. On this figure, the stream A ($S(S,208,197)$) is shown as a dotted line. A complex gun $Gb(E,179,200)$ creates a complementary duplicate stream $\overline{A}$ towards the East. The two outputs are redirected by complex guns $Gb(W,253,142)$, $Gb(S,212,65)$ and $Ga(S,273,262)$ until they are

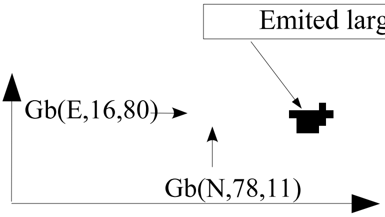


Figure 47. Schematics of a large glider gun $L(E,16,80)$, made of two complex Gb guns, with a expulsed glider at the right.

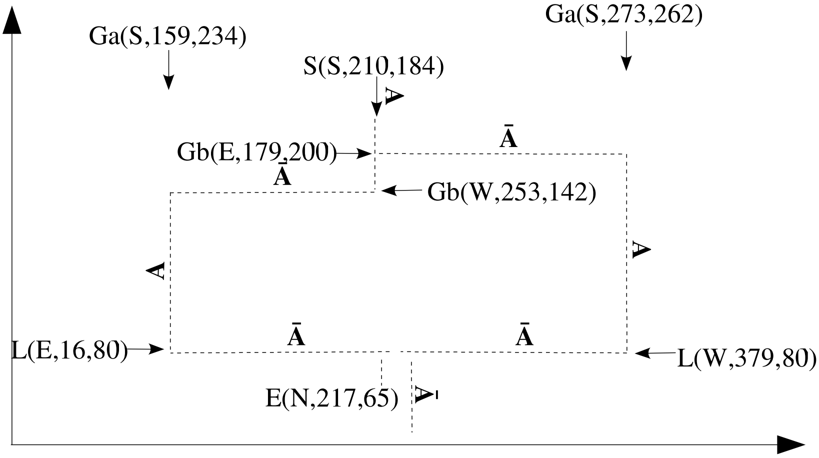


Figure 48. Complementation of stream A: the analytical representation for the NOT gate is $\{L(E,$ 16, 80), $Gb(S,$ 212, 65), $Gb(W,$ 241, 36), $Gb(W,$ 253, 142), $L(W,$ 379, 80), $Ga(S,$ 159, 234), $Gb(E,$ 179, 200), $Ga(S,$ 273, 262), $S(S,$ l, 210, 184), $E(N,$ 219, 78)$\}$.

vertical again. Then, they are complemented into large gliders by the two guns of large gliders $L(E,16,80)$ and $L(W,379,80)$. When the frontal collision between the two streams of large gliders occurs, a complementary stream $\overline{A}$ is created towards the same direction as the original A stream while the other one is "eaten" by $E(N,216,65)$.

5.4. Simulation of One Cell of the Game of Life

A single cell of the Game of Life can be implemented as a boolean function computing the value of a cell S at generation $n+1$ from its value at the generation n and the ones of its eight neighbours $C_1 \ldots C_8$ at generation n.

The rules of the Game of Life are the following: a "living" cell dies at the next generation unless it has two or three neighbours. A dead cell comes alive at the next generation iff it has three neighbours in the current generation.

Supposing that the addition of $C_1 + \ldots + C_8$ gives a four bit number $n_3 n_2 n_1 n_0$. Eight neighbours at the state 1 give the number 1000 and the bit n_3 is equal to 1 just for this case. A cell at state 0 or 1 surrounded by eight neighbours becomes a cell at state 0 at the next generation. If the bit n_3 is ignored so eight neighbours give the number 000 and the value of a cell surrounded by zero neighbour is always 0 at the next generation. So the bit n_3 can be ignored.

The rules of the Game of Life can be simply expressed by the formula $S_{n+1} = \overline{n_2}.n_1.(S_n + n_0)$, which can be translated into a combination of $NAND$ gates. This function, implementing a cell of the Game of Life, is implemented in the figure 49.

5.5. Simulation of the Game of Life

In order to simulate the Game of Life, one must first find in R a simulation of a cell of the Game of Life, and then a way to tile a surface with any number of interconnected cells.

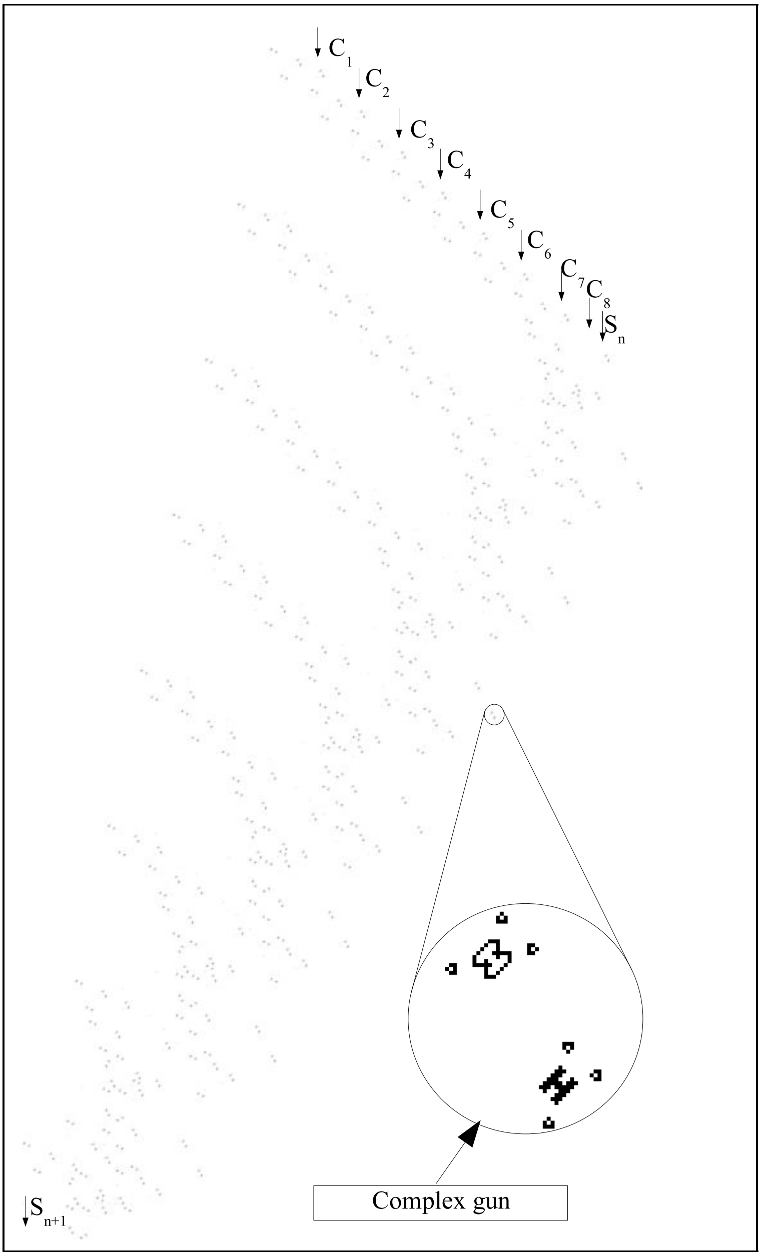


Figure 49. Simulation of a cell of the Game of Life by *R*.

In order to simulate the Game of Life in *R*, proof must now be given that it is possible to tile a surface of identical cells, each interconnected with their 8 neighbours. In order to interconnect a cell with their neighbours, proof must be given that two streams in any position can be redirected in order to become the input streams of any circuit. It means the synchronisation and the intersection of stream must be possible that is the subject of the first subsections. The last subsection shows the tile of the plan of the Game of Life.

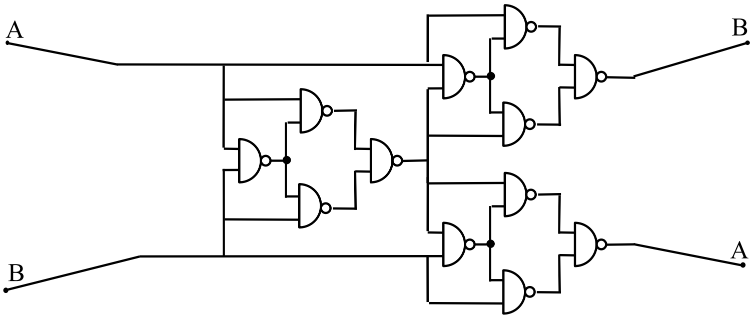


Figure 50. Cuning combinaison of Nand Gate that realises a stream intersection without interference.

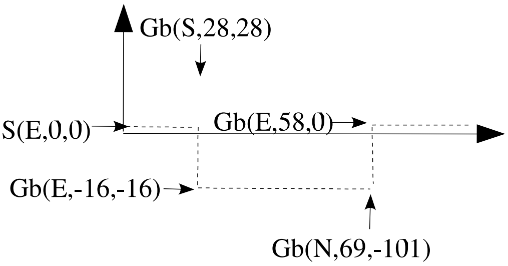


Figure 51. Stream temporisation for synchronisation purposes.

5.5.1. Intersection of Streams

Thanks to complex guns, gliders in a stream are separated by 45 cells. This means that it is possible to have two streams cross each other without any interference. If, for a synchronisation reason, interference cannot be avoided, [4] shows how a stream intersection can be realised without interference by an cunning combination of *NAND* gates as shown in the figure 50.

5.5.2. Synchronisation

It is important to be able to delay one stream w.r.t another, in order to synchronise them properly just before they enter a logic gate, for instance. This can be done precisely by diverting four times the stream to be delayed with orthogonal guns (cf. figure 51).

5.5.3. Simulation of the Game of Life in *R*

A single cell of the Game of Life can be implemented as a boolean function computing the value of a cell S at generation $n+1$ from the value of its eight neighbours $C_1 \ldots C_8$ at generation n.

All cells being identical, the inputs of a cell must physically correspond to the outputs of its neighbours. Therefore, the way a cell receives the state of its neighbours can be induced from the way it sends its own state to its neighbours, which is what is described below and in figure 52.

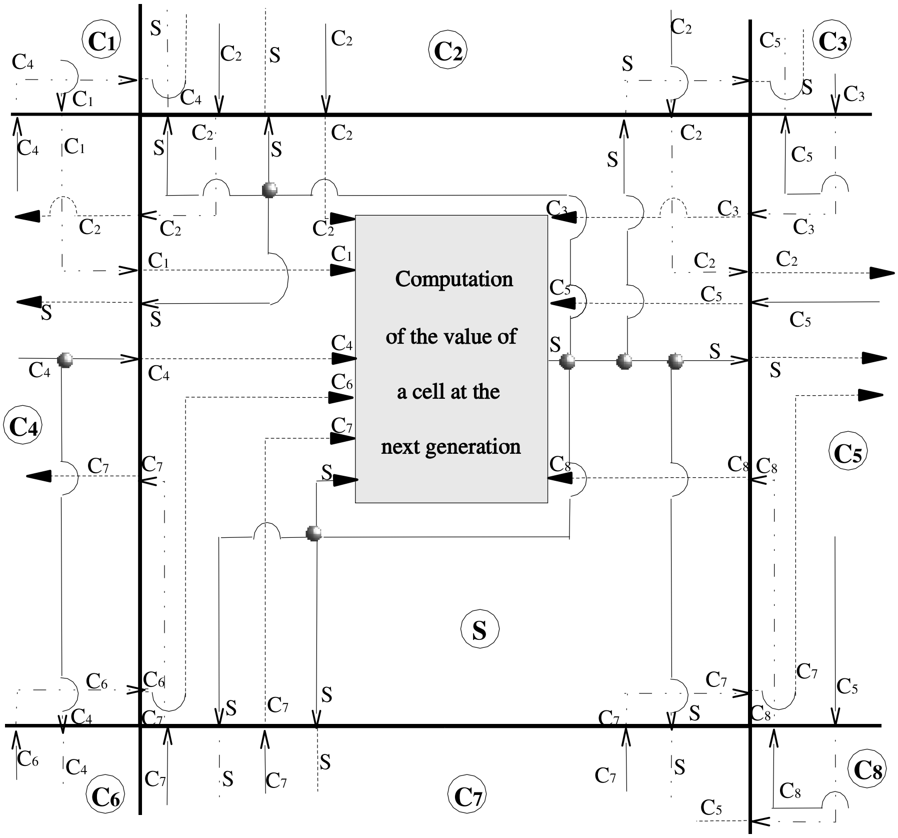


Figure 52. Diagram of the tiling of the plan of the Game of Life.

It is straightforward for a cell to send its state to its cardinal neighbours C_2, C_4, C_5, C_7. Sending its state to neighbours C_1, C_3, C_6, C_8 is however more tricky, since those neighbours are situated diagonally. This is done by passing the information to their neighbours C_2 and C_7. Therefore, one can see in figure 52 that the state S of the cell is sent three times to C_2 and three times to C_7, so that C_2 (resp. C_7) can keep one stream for its own use, and pass the two others to its horizontal neighbours, C_1 and C_3 (resp. C_6 and C_8).

S being itself a top neighbour of cell C_7, one sees how the state of C_7 is passed over to C_4 and C_5 in the same way that C_2 will pass over the information of the state of S to C_1 and C_3.

6. Conclusion

This paper deals with the emergence of computation in complex systems with local interactions. Evolutionary search methods are used to find gliders. The algorithm succeeded in finding thousands of gliders.

Glider guns appear in the evolution of a random configuration of cells by the transition rule of one of the automaton. The cellular automaton called R was picked up as a potential candidate for a universal automaton.

The demonstration of universality of R is inspired by the demonstration of universality of the Game of Life. However a NAND gate can not be simulated by the same method. The demonstration of universality of R used a complicated combinaison of guns to simulate a NAND gate. A cell of the Game of Life is simulated by R then the plan of the Game of Life is tiled by identical simulation of cells in order to prove the universality of R.

An extensive bibliographic research seems to show that the automaton R is the first another 2D 2 state dynamical universal (in the Turing sense) automaton, other than the famous the Game of Life, discovered in I.

The proof of universality of the automaton R can be generalized on other automata accepting glider guns. Some discovered automata accept glider guns thus they are potential candidates for universal automata. Therefore this research provides an element of the answer to the frequence of universal automata related to emergence of computation in complex systems with simple local behaviour.

Further goals are now to find whether other universal automata than the Game of Life and R exist, and how common they are. Then, another domain that seems worth exploring is how this approach could be extended to automata with more than 2 states or more than two dimensions.

Finally, the study of the construction of an automatic system of selection / discovery of this type of automata based on evolutionary algorithms is far more interesting. The discovery of universal cellular automata can lead to a new classification of cellular automata.

Future work could also be to calculate for each automaton some rule-based parameters, e.g., Langton's lamda [5]. All automata exhibing glider guns may have similar values for these parameters that could lead to a better understanding of the link between the rule transition and the emergence of computation in cellular automata and therefore the emergence of computation in complex systems with simple components.

References

[1] S. Wolfram. Universality and complexity in cellular automata. *Physica D*, **10**:1–35, 1984.

[2] E. R. Banks. *Information and transmission in cellular automata* . PhD thesis, MIT, 1971.

[3] S. Wolfram. *Twenty problems of the theory of cellular automata*. Physica Scripta, 170–183, 1985.

[4] A. Dewdney. *The planiverse*. Pseidon Press, 1984.

[5] C. L. Langton. *Computation at the edge of chaos*. Physica D, Vol. 42, 1990.

[6] E. Berlekamp, J. H. Conway, and R. Guy. Winning ways for your mathematical plays. *Academic Press, New York*, 1982.

[7] C. Bays. Candidates for the game of life in three dimensions. *Complex Systems*, **1**:373–400, 1987.

[8] E. Sapin, O. Bailleux, and J.J. Chabrier. Research of complex forms in the cellular automata by evolutionary algorithms. *EA03. Lecture Notes in Computer Science*, **2936**:373–400, 2004.

[9] E. Sapin, O. Bailleux, and J.J. Chabrier. Research of a cellular automaton simulating logic gates by evolutionary algorithms. *EuroGP03. Lecture Notes in Computer Science*, **2610**:414–423, 2003.

[10] N. Margolus. Physics-like models of computation. *Physica D*, **10**:81–95, 1984.

[11] K. Lindgren and M. Nordahl. Universal computation in simple one dimensional cellular automata. *Complex Systems*, **4**:299–318, 1990.

[12] K. Morita, Y. Tojima, I. Katsunobo, and T. Ogiro. Universal computing in reversible and number-conserving two-dimensional cellular spaces. *In A. Adamatzky (ed.), Collision-Based Computing, Springer Verlag.*, pages 161–199, 2002.

[13] A. Adamatzky. Universal dymical computation in multi-dimensional excitable lattices. *International Journal of Theoretical Physics*, **37**:3069–3108, 1998.

[14] E. Sapin, O. Bailleux, J.J. Chabrier, and P. Collet. Demonstration of the universality of a new cellular automaton. *IJUC*, **2**(3), 2006.

[15] M. Mitchell, J. P. Crutchfield, and P. T. Hraber. Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, **75**:361–391, 1994.

[16] M. Mitchell, P. T. Hraber, and J. P. Crutchfield. Revisiting the edge of chaos : Evolving cellular automate to perform computations. *Complex systems*, **7**:89–130, 1993.

[17] S. Wolfram. *A New Kind of Science*. Wolfram Media, Inc., Illinois, USA, 2002.

[18] J. Von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, Ill., 1966.

[19] E. Sapin. http://uncomp.uwe.ac.uk/sapin/gun.zip, 2009.

[20] M. Sipper. Evolution of parallel cellular machines. *D. Stauffer, editor, Annual Reviews of Computational Physics*, V. World Scientific:243–285, 1997.

[21] W. Hordijk, J. P. Crutchfield, and M. Mitchell. Mechanisms of emergent computation in cellular automata. *Parallel Problem Solving from Nature-V, A. E. Eiben, T. Bck, M. Schoenauer, and H.-P. Schwefel (eds.), Springer-Verlag*, **866**:344–353, 1998.

[22] G. J. Martinez, A. Adamatzky, and H. V. McIntosh. Phenomenology of glider collisions in cellular automaton rule 54 and associated logical gates chaos. *Fractals and Solitons*, **28**:100–111, 2006.

[23] A. Wuensche. Discrete dinamics lab (ddlab),www.ddlab.org, 2005.

[24] D. Eppstein. http://www.ics.uci.edu/ eppstein/ca/.

[25] N. H. Packard. Adaptation toward the edge of chaos. *In J. A. S. Kelso, A. J. Mandell, M. F. Shlesinger, eds., Dynamic Patterns in Complex Systems*, pages 293–301, 1988.

[26] M. Resnick and B. Silverman. Exploring emergence, http://llk.media.mit.edu/projects/emergence/, 1996.

[27] J.D. Lohn and J.A. Reggia. Automatic discovery of self-replicating structures in cellular automata. *IEEE Transactions on Evolutionary Computation* , **1**:165–178, 1997.

[28] J.J. Ventrella. A particle swarm selects for evolution of gliders in non-uniform 2d cellular automata. *In Artificial Life X: Proceedings of the 10th International Conference on the Simulation and Synthesis of Living Systems* , pages 386–392, 2006.

[29] E. Sapin, O. Bailleux, and J. Chabrier. Research of complexity in cellular automata through evolutionary algorightms. *Complex systems,* **17**.

[30] E. Sapin and L. Bull. A genetic algorithm approach to searching for glider guns in cellular automata. *IEEE Congress on Evolutionary Computation*, pages pp. 2456–2462.

[31] J. Uras, R. Rechtman, and A. Enciso. Sensitive dependence on initial conditions for cellular automata. *Chaos: An Interdisciplinary Journal of Nonlinear Science* , Vol 7, Issue 4:pp. 688–693, 1997.

[32] J. Kennedy and C. Eberhart. Particle swarm optimization. *Proceedings of the 1995 IEEE Conference on Neural Networks*, 1995.

[33] D. Wolz and P.B. de Oliveira. Very effective evolutionary techniques for searching cellular automata rule spaces. *Journal of Cellular Automata*, Vol 3, Issue 4:pp. 289–312, 2008.

[34] A. Wuensche and A. Adamatzky. On spiral glider-guns in hexagonal cellular automata: activator-inhibitor paradigm. *International Journal of Modern Physics C* , Vol 17, No. 7:pp. 1009–1026, 2008.

[35] M.M. Waldrop. *Complexity: The Emerging Science at the Edge of Chaos. (New York: Simon and Schuster*. Simon and Schuster, New York, NY, 1992.

[36] A. Ilachinski. *Cellular Automata*. World Scientific, 1992.

[37] N. Ollinger. Universalities in cellular automata a (short) survey. *In B. Durand, editor, Symposium on Cellular Automata Journees Automates Cellulaires (JAC08)* , pages pp. 102–118, 2008.

[38] M. Delorme. An introduction to cellular automata. *In: M. Delorme and J. Mazoyer (Editors) Cellular Automata: A Parallel Model (Kluwer)* , pages pp. 5–51, 1999.

[39] P. Rendell. Turing universality in the game of life. *In Adamatzky, Andrew (ed.), Collision-Based Computing, Springer* , pages pp. 513–539, 2002.

[40] E. Sapin and L. Bull. Searching for glider guns in cellular automata: Exploring evolutionary and other techniques. *EA07. Lecture Notes in Computer Science*, **4926**:255–265, 2007.

[41] Aristotle. Metaphysics. Book 8.6.1045a:8-10.

[42] J. Stuart Mill. On the Composition of Causes. A System of Logic Ratiocinative and Inductive (1872 ed.), London: John W. Parker and Son. pp:371, 1843.

[43] J. S. Huxley and T. H. Huxley. Evolution and Ethics: 1893-1943. London, 1947: The Pilot Press, p. 120, 1947.

[44] G. Terrazas and P. Siepmann and G. Kendall and N. Krasnogor. An evolutionary methodology for the automated design of cellular automaton-based complex systems. *Journal of Cellular Automata*, Vol 2:pp. 77–102, 2007.

[45] K. M. Evans. Is Bosco's Rule Universal? *LNCS* **3354**, (2005), 188-199.

In: Cellular Automata
Editor: Thomas M. Li, pp.249-272

ISBN: 978-1-61761-592-4

Chapter 10

A Numerical Implementation of an Encryption System of Compressed Signals with a Cellular Automata Approach

J. S. Murguía[a,b] *, ***M. Mejía–Carlos***[a], ***H. C. Rosu***[b], **and G. Flores-Eraña**[a]
[a] UASLP, Universidad Autónoma de San Luis Potosí
San Luis Potosí, S.L.P., México
[b] IPICyT, Instituto Potosino de Investigacion Cientifica y Tecnologica
San Luis Potosí, S.L.P., México

Abstract

This work presents a numerical implementation of a system that considers an encryption process of signals compressed by means of the Haar wavelet transform. An evaluation of the pseudorandom generator used in the encryption scheme, which is based on a cellular automaton, is carried out with some variants. In addition, the multifractal properties of the representative matrix of the generator are discussed.

PACS 05.40.-a., 05.45.-a., 05.45.Tp.

Keywords: Cellular automata, Compression, Encryption system, Haar wavelet transform, Multifractal spectrum, Pseudo-random generator.

1. Introduction

Different fields, such as electrical engineering, physics, and mathematics, have shown a growing interest in protecting and manipulating huge amounts of data. With the recent advances in technology, the processing of different kinds of information is a source of interest to many people having as their main goals transferring or storing the data in a secure way. Thus, there is still a driving need to look for efficient algorithms to compress and efficiently and safely transmit the processing of different kinds of data.

*E-mail address: ondeleto@uaslp.mx

On the one hand, different compression schemes, which can either be lossless or lossy, work by squeezing redundancy out of data, reducing substantially the initial size of the analyzed signals. In these issues, the wavelet transform has proved to be a powerful tool to efficiently process signals that involve large amounts of information. In particular, it has been noticed that this transform is a flexible mathematical tool employed in a great variety of applications and its numerical implementation is often easy to perform [16].

On the other hand, to maintain the information protected, several cryptosystems have been proposed using the encryption-decryption processes. The requirements to fulfill the security needs of different kind of signals have led to the development of good encryption techniques. [1, 4].

Actually, there exists a large number of encryption systems whose main objective is to protect information through an algorithm that makes use of one or more keys.

2. Elementary Cellular Automata

The elementary cellular automata (ECA) can be considered as discrete dynamical systems that evolve in discrete time steps. The state space of a CA of size N is the set $\Omega = \mathbb{Z}_k^N$ of all sequences of N cells that take values from $\mathbb{Z}_k = \{0, 1, \ldots, k-1\}$, where its evolution is defined by the repeated iteration of an evolution operator $\mathcal{A} : \mathbb{Z}_k^N \to \mathbb{Z}_k^N$. In this paper, we consider $k = 2$ where $\mathbb{Z}$ is the set of integers. An automaton state $\underline{x} \in \mathbb{Z}_2^{\mathbb{Z}}$ has coordinates $(\underline{x})_i = x_i \in \mathbb{Z}_2$ with $i \in \mathbb{Z}$, and the automaton state at time $t \geq 0$ is denoted by $\underline{x}^t \in \mathbb{Z}_2^{\mathbb{Z}}$ and its evolution is defined iteratively by the rule $\underline{x}^{t+1} = \mathcal{A}(\underline{x}^t)$. Starting from the initial state $\underline{x}^0$, the automaton generates the forward space-time pattern $\mathbf{x} \in \mathbb{Z}_2^{\mathbb{Z}\times\mathbb{N}}$ with state $(\mathbf{x})^t = \underline{x}^t = \mathcal{A}^t(\underline{x}^0)$ reached at from $\underline{x}^0$ after $t \in \mathbb{N}$ time steps. $\mathbb{N}$ denotes the set of nonnegative integers.

One can see that the time, space, and states of this system take only discrete values. The ECA considered evolves according to the local rule $x_i^{t+1} = \mathcal{A}_L(x_{i-1}^t, x_i^t, x_{i+1}^t) = [x_{i-1}^t + x_i^t] \mathrm{mod}\ 2$, which corresponds to the rule 90. The following is the lookup table of rule 90.

Number	7	6	5	4	3	2	1	0
Neighborhood	111	110	101	100	011	010	001	000
Rule result	0	1	0	1	1	0	1	0

The third row shows the future state of the cell if itself and its neighbors are in the arrangement shown above in the second row. In fact, a rule is numbered by the unsigned decimal equivalent of the binary expression in the third row. When the same rule is applied to update cells of ECA, such ECA are called uniform ECA; otherwise the ECA are called non-uniform or hybrids. It is important to observe that the evolution rules of ECA are determined by two main factors, the rule and the initial conditions.

3. Encryption System

We consider the encryption scheme used in [2], where the synchronization phenomenon of cellular automata has been applied to devise the two families of permutations and an asymptotically perfect pseudorandom number generator. In fact, Mejía and Urías [3] constructed an ergodic and mixing transformation of binary sequences in terms of a cellular automaton, which is the basic element of the pseudorandom generator number. The class of block cryptosystem considered transforms a plain text sequence $\mathbf{m}$ to a sequence $\mathbf{c}$, called the ciphertext. The transformation $\mathbf{m} \mapsto \mathbf{c}$ is selected from an indexed family of permutations $\Psi = \{\psi_{\mathbf{k}} : M \to C | \mathbf{k} \in K\}$ by choosing an index $\mathbf{k}$ from the set of indices K. The sets M, C and K are all sets of binary words of length N, i.e., Z_2^N, where $Z_2 = \{0, 1\}$. The words in M and C are called the *clearblocks* and *cipherblocks*, respectively, whereas the words in the set of indices K are the *encyphering keys*. To disclose from the sequence of cipherblocks, the cryptosystem also provides the family of inverse permutations $\Phi = \{\phi_{\mathbf{k}} : C \to M | \mathbf{k} \in K\}$ such that for every $\mathbf{k} \in K$ one has $\mathbf{m} = \phi_{\mathbf{k}}(\psi_{\mathbf{k}}(\mathbf{m}))$. In this process, we demand to know the seed that was used to generate the pseudorandom sequence of keys, i.e., the encryption and decryption processes use the same deterministic generator that is initialized with a common seed. Notice that the plain text to be encrypted is generally much longer than the length N that is accepted by the family of permutations Ψ. In this case, we proceed to divide it into succession of blocks $\mathbf{m}^0$, $\mathbf{m}^1$, $\mathbf{m}^2, \ldots$ each of length N, and these blocks are then encrypted sequentially by using a different key $\mathbf{k}^i$ for each block $\mathbf{m}^i$. If the cipher text is intercepted, this encryption system must avoid that the intruder be able to infer any information about the text. In this case it is relevant to select as random as possible the succession of permutations, because the intruder must have to agree on a very long sequence of keys that determines the permutations [2]. This problem is solved by using a pseudorandom generator of keys. Figure 1 illustrates the complete encryption scheme.

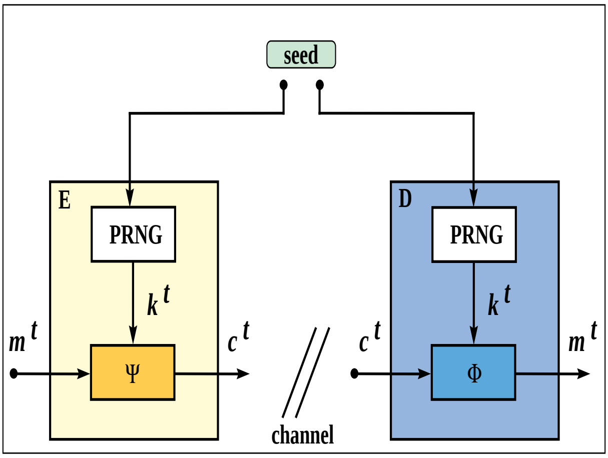


Figure 1. The complete encryption scheme with its main components: the indexed families of permutations and the pseudorandom generator of keys.

In order to explain how the main components were implemented in this encryption system, we briefly describe some necessary concepts.

3.1. Synchronization in Cellular Automata

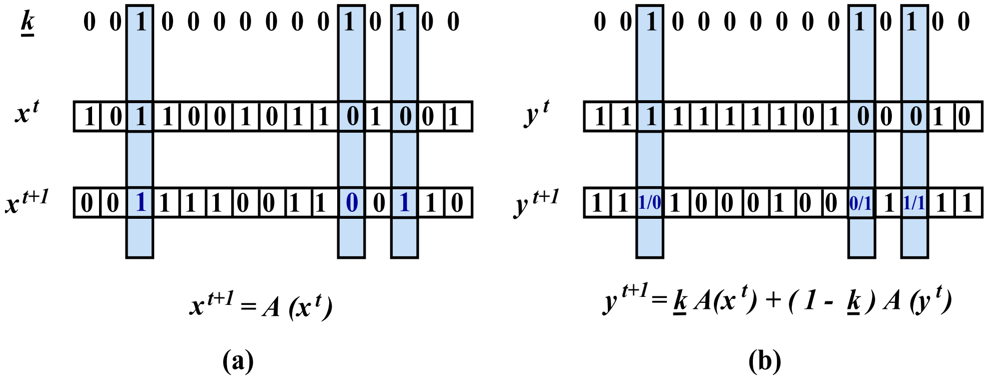
Before describing the phenomenon of synchronization in coupled linear ECA, we require the following concept.

3.1.1. Unidirectional coupling

The unidirectional coupling of two cellular automata can be implemented when the particular set of coupled coordinates is copied from the driver cellular automata to the replica cellular automata. We illustrate the coupling of two CA in a drive/response configuration through the following situation. First, a set of coupling coordinates is specified by means of a coupling sequence $\underline{\mathbf{k}} = (k_0, k_1, \ldots, k_{N-1})$, such that $k_i = 1$ if the i-th coordinate is a coupling coordinate, and $k_i = 0$ otherwise. As a result, the sequence $\underline{\mathbf{k}}$ determines which coordinates from the driver's configuration are to be copied onto the replica's configuration. Now, suppose that the evolution operator is $\mathcal{A}$ and let the initial states of the driver and the response systems be $\underline{x}^0$ and $\underline{y}^0$, respectively. If the states at time t are $\underline{x}^t$ and $\underline{y}^t$, respectively, then the state of the driver CA at time $(t+1)$ is $\mathcal{A}(\underline{x}^t)$, and the state of the response CA at time $(t+1)$ is

$$\underline{y}^{t+1} = (\underline{\mathbf{1}} - \underline{\mathbf{k}})\mathcal{A}(\underline{y}^t) + \underline{\mathbf{k}}\mathcal{A}(\underline{x}^t), \tag{1}$$

where $\underline{\mathbf{1}} = (1, 1, \ldots, 1)$ is the vector of size N whose entries are all 1's. Addition and multiplication of vectors in (1) are performed coordinate-wise. Two CA coupled unidirectionally in this manner constitute a *coupled pair* $(\mathcal{A}, \underline{\mathbf{k}})$ in a drive/response configuration. Figure 2 illustrates an unidirectional coupling with $N = 15$, where $\mathcal{A}$ corresponds to the evolution of the rule 90, and the coupling sequence $\underline{\mathbf{k}}$ has the coupled coordinates k_2, k_{10}, and k_{12}.



$\underline{k}$	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0
x^t	1	0	1	1	0	0	1	0	1	1	0	1	0	0	1
x^{t+1}	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0

$x^{t+1} = A\,(x^t)$

(a)

	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0
y^t	1	1	1	1	1	1	1	1	0	1	0	0	0	1	0
y^{t+1}	1	1	1/0	1	0	0	0	1	0	0	0/1	1	1/1	1	1

$y^{t+1} = \underline{k}\,A(x^t) + (1 - \underline{k})\,A\,(y^t)$

(b)

Figure 2. A coupled pair of CA with $N = 15$. (a) Driver CA that evolves autonomously from x^t to x^{t+1}. (b) Replica CA that evolves from y^t to y^{t+1}. In this example, there are three coordinates of the driver coupled onto the replica.

3.1.2. Synchronization

A coupled pair $(\mathcal{A}, \underline{\mathbf{k}})$ synchronizes when the difference between the two state vectors $\underline{x}^t$, $\underline{y}^t$, corresponding to the driver and the replica CA, respectively, eventually equals the null vector $\underline{\mathbf{0}} = (0, 0, \ldots, 0)$ after a certain number of steps at time t. In Reference [1] it has been shown that a pair of linear ECA, where the local rule $\mathcal{A}_{\mathcal{L}}$ corresponds to the automaton rule 90, synchronizes if every pair of consecutive coupled coordinates are separated by a block of $N = 2^k - 1$ sites uncoupled. In the coupled system the configuration of the driver evolves autonomously under local rule $x_i^{t+1} = \mathcal{A}_{\mathcal{L}}(x_{i-1}, x_i, x_{i+1})$, while the configuration of the replica cellular automata in each coupled coordinate i evolves according to $y_i^{t+1} = x_i^{t+1}$ and each uncoupled coordinate i follows the local rule, $y_i^{t+1} = \mathcal{A}_{\mathcal{L}}(y_{i-1}, y_i, y_{i+1})$ [1].

As was pointed out in [1, 2], the system consisting of the evolution automaton rule 90 $\mathcal{A}_L$ acting on binary sequences of length $N = 2^k - 1$ subjected to fixed boundary condition reaches, at a time $t < 2^k$, an orbit that depends only on the boundary and is independent of the initial configuration. This property is crucial to construct the main components of the encryption system. For more details see [1, 2].

3.2. The Basic Unit Cipher

With the synchronization phenomenon of CA, it is possible to implement in a flexible way the cryptography primitives, the pseudorandom generator of keys (function h) and the indexed families of permutations Ψ and Φ. To construct these primitives we will take into account the following fact. Let $\underline{x}^0$ be an initial infinite sequence, i.e.,

$$\underline{x}^0 = (\ldots, x_{-1}^0, x_0^0, x_1^0, \ldots, x_{N-1}^0, x_N^0, \ldots) \tag{2}$$

that evolves according the local rule $\mathcal{A}_L(x_{i-1}^t, x_i^t, x_{i+1}^t) = [x_{i-1}^t + x_i^t] \bmod 2$, from $t = 0$ to $t = N = 2^k - 1$, where $i \neq 0$ and $i \neq N + 1$, since x_0^t and x_{N+1}^t are externally assigned at each time t.

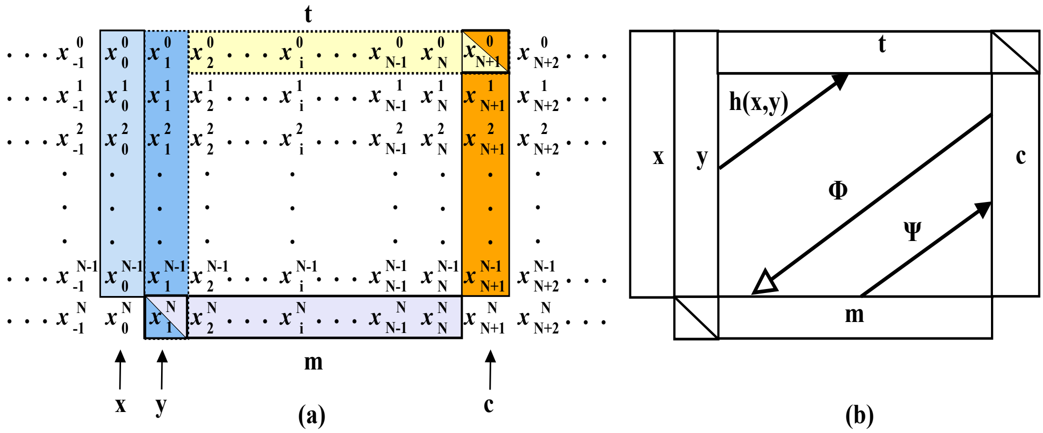

t
. . . x^0_{-1} x^0_0 x^0_1 x^0_2 . . . x^0_i . . . x^0_{N-1} x^0_N x^0_{N+1} x^0_{N+2} . . .
. . . x^1_{-1} x^1_0 x^1_1 x^1_2 . . . x^1_i . . . x^1_{N-1} x^1_N x^1_{N+1} x^1_{N+2} . . .
. . . x^2_{-1} x^2_0 x^2_1 x^2_2 . . . x^2_i . . . x^2_{N-1} x^2_N x^2_{N+1} x^2_{N+2} . . .
. . . x^{N-1}_{-1} x^{N-1}_0 x^{N-1}_1 x^{N-1}_2 . . . x^{N-1}_i . . . x^{N-1}_{N-1} x^{N-1}_N x^{N-1}_{N+1} x^{N-1}_{N+2} . . .
. . . x^N_{-1} x^N_0 x^N_1 x^N_2 . . . x^N_i . . . x^N_{N-1} x^N_N x^N_{N+1} x^N_{N+2} . . .
m
x
y
c
(a)
t
h(x,y)
x
y
c
Φ
Ψ
m
(b)

Figure 3. (a) Space-time pattern from an infinite initial state according to the evolution of the rule 90, $\mathcal{A}_L(x_{i-1}^t, x_i^t, x_{i+1}^t) = [x_{i-1}^t + x_i^t] \bmod 2$. (b) Primitives defined by the basic unit cypher. The functions h and Ψ are determined by iterating the CA backward in time, whereas the function Φ is computed by running the CA forward in time.

Figure 3 (a) shows the space-time pattern from the infinite initial state (2), according to the evolution of the automaton rule. From the coupled coordinates, x_0^0 and x_{N+1}^0, we define the basic unit cipher (BUC) as the $N \times N$ square pattern in the lattice that consists of the N time-running words $(x_1^0, x_1^1, \ldots, x_1^N), \ldots, (x_N^1, x_N^2, \ldots, x_N^N)$. The first time-running word is distinguished by the name $\underline{k}' = (x_1^1, x_1^2, \ldots, x_1^N)$. The words surrounding the square are $\mathbf{x} = (x_0^0, x_0^1, \ldots, x_0^{N-1})$ on the left side, $\mathbf{c} = (x_{N+1}^1, x_{N+1}^2, \ldots, x_{N+1}^N)$ on the right side, $\mathbf{t} = (x_1^1, x_2^1, \ldots, x_N^1)$ on the top, and $\mathbf{m} = (x_N^{N+1}, x_N^{N+1}, \ldots, x_N^{N+1})$ at the bottom.

The family of permutations $\mathbf{\Psi}$ and $\mathbf{\Phi}$, and the h function, are defined with the help of the BUC, where we can identify the five main words $\mathbf{x}$, $\mathbf{y}$, $\mathbf{m}$, $\mathbf{c}$ and $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$.

Permutation $\mathbf{c} = \Psi_{\mathbf{x}}(\mathbf{m})$ The word located on the right side of the BUC, $\mathbf{c} = (x_{N+1}^0, x_{N+1}^1, \ldots, x_{N+1}^{N-1})$, is a cipher-block word; it is obtained using the indexed family permutation $\Psi_{\mathbf{x}}$, i.e., $\mathbf{c} = \Psi_{\mathbf{x}}(\mathbf{m})$. This permutation determine the cipher-blocks, when the ECA is iterated backward in time using the input words $\mathbf{x}$ and $\mathbf{m}$.

Permutation $\mathbf{m} = \Phi_{\mathbf{x}}(\mathbf{c})$ The word $\mathbf{m}$ is a plain text sequence, and is localized at the bottom of the BUC, $\mathbf{m} = (x_1^N, x_2^N, \ldots, x_N^N)$. To compute the inverse permutation $\mathbf{m} = \Phi_{\mathbf{x}}(\mathbf{c})$, i.e., to bring $\mathbf{c}$ back to $\mathbf{m}$, the automaton is made to run forward in time, using the input words $\mathbf{c}$ and $\mathbf{x}$.

Function $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$ The two words located on the left side of the BUC, $\mathbf{x}$ and $\mathbf{y}$, are the input of the function $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$. These words are $\mathbf{x} = (x_0^0, x_0^1, \ldots, x_0^{N-1})$, and $\mathbf{y} = (x_1^0, x_1^1, \ldots, x_1^N)$. The result of function h is on the top of the BUC and is identified as $\mathbf{t} = (x_2^0, x_3^0, \ldots, x_{N+1}^0)$. To generate this function, the automaton is also iterated backwards in time, using as input words $\mathbf{x}$ and $\mathbf{y}$.

The objects implemented in the BUC are shown in Figure 3(b). Notice that the sequences of $\mathbf{y}$ and $\mathbf{m}$ share the symbol x_1^N, whereas $\mathbf{t}$ and $\mathbf{c}$ the symbol x_{N+1}^0. The backward evolution of the ECA is illustrated in Figure 4, which is employed as an operation to devise the permutation Ψ and function $\mathbf{t} = h$.

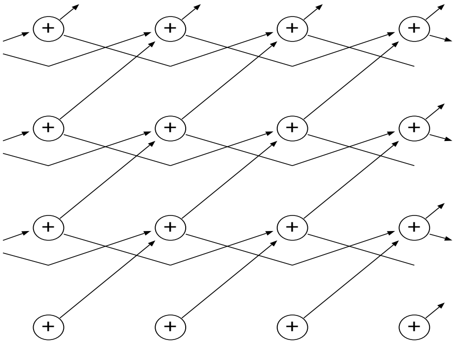


Figure 4. Backward evolution of the ECA.

4. Pseudo Random Sequences Generator

To implement numerically the PRNG in its basic form, we follow its algorithm, which is shown in Fig. 5. At first, the key generator requires two seeds, $\mathbf{x} = \mathbf{x}_0^{k1}$, of N bits, and $\mathbf{y} = \mathbf{x}_0^k$, of $(N+1)$ bits, which are the input of the function $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$. Considering the seeds $\mathbf{x} = (x_0^0, x_0^1, \ldots, x_0^{N-1})$ and $\mathbf{y} = (x_1^0, x_1^1, \ldots, x_1^N)$, thus, the first number generated of N bits is the sequence output of function h, $\mathbf{t} = \{t_1, t_2, t_3, \ldots, t_N\}$. Now, this sequence is fed back to the input, which becomes the next value of $\mathbf{x}$, and the previous value of $\mathbf{x}$ becomes the initial bits of the new $\mathbf{y}$, where the missing bit is the least significant bit (LSB) of the previous $\mathbf{y}$, which becomes the most significant bit (MSB) of this sequence, and the same procedure is iterated repeatedly.

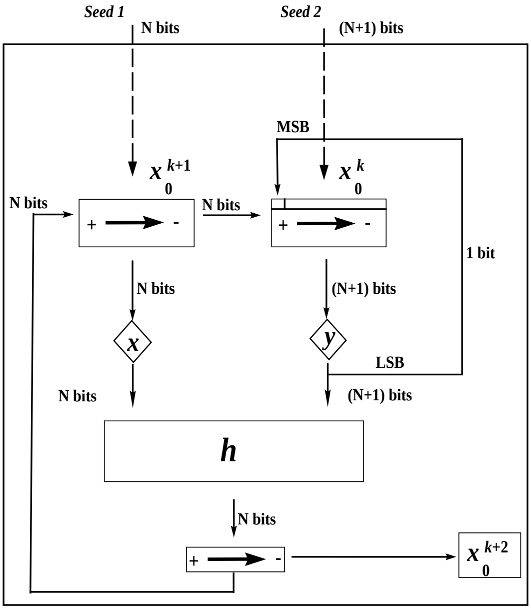


Figure 5. Basic form of the pseudo-random number generator. MSB and LSB correspond to the most significant bit and the least significant bit, respectively.

As was said above, in order to compute the function $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$ it is required that the cellular automaton runs backward in time. Such situation is depicted in Figure 6, where the symbol of a circled $+$ represents a XOR gate and the connectivity of gates follows the automaton rule. However, this way to compute the pseudo-random sequences is not efficient since it requires the application of the local rule of the automaton at all points in a lattice of the order of N^2, where N is the number of bits considered in the generation process.

To overcome this, Mejía and Urías [3] formulated an efficient algorithm that gets rid of the intermediate variables and produces Boolean expressions for the coordinates of the output sequence $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$ in terms of the input $(\mathbf{x}, \mathbf{y})$. This algorithm offers a Boolean representation of h, without intermediate steps, in terms of some “triangles” in the underlying lattice.

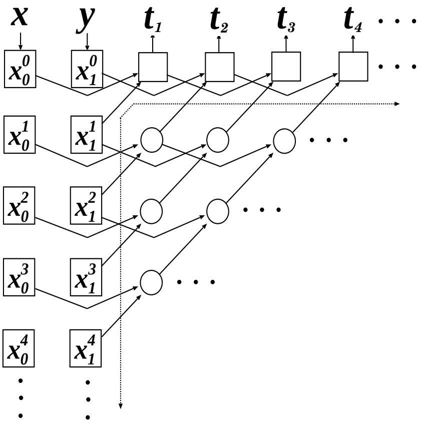


Figure 6. Generation of a pseudo-random key with input $(\mathbf{x}, \mathbf{y})$ and output $\mathbf{t} = h(\mathbf{x}, \mathbf{y})$.

With the aim of computing the pseudo random sequences of N bits, in Reference [15] the sequence matrix $\mathbf{H}_N$ has been introduced.

This matrix has dimensions of $(2N + 1) \times (2N + 1)$ and is formed by the matrices $\mathbf{H}_{N_t}$ and $\mathbf{H}_{N_b}$, which constitute the top and bottom parts of $\mathbf{H}_N$, that is, $\mathbf{H}_N = \big(\mathbf{H}_{N_t}; \mathbf{H}_{N_b}\big)$. The matrix $\mathbf{H}_{N_t}$ has dimensions of $N \times (2N + 1)$ elements and it is generated initially from two vectors, $\mathbf{v} = [v_1, 0, \ldots, 0, v_{N+2}, \ldots, 0]$ and $\mathbf{w} = [0, w_2, 0, \ldots, w_{N+1}, 0, w_{N+3}, \ldots, 0]$, where the components $v_1, v_{N+2}, w_2, w_{N+1}$ and w_{N+3} have a value of 1, and N is the number of bits, i.e., $\mathbf{v}$ and $\mathbf{w}$ are vectors with $(2N+1)$ elements. The vectors $\mathbf{v}$ and $\mathbf{w}$ constitute the two first rows of the matrix $\mathbf{H}_{N_t}$ and the other $(N-2)$ rows are generated by applying an addition modulo 2 operation of the two previous rows, with the elements of the previous row shifted to the right by one position. The matrix $\mathbf{H}_{N_b}$ has dimensions $(N + 1) \times (2N + 1)$, and it is composed by an identity matrix in the first $(N + 1)$ columns and zeros in the rest. As an example for $N = 7$, we have that the top and bottom matrices are the following:

$$\mathbf{H}_{7_t} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \qquad (3)$$

$$\mathbf{H}_{7_b} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4}$$

then the matrix $\mathbf{H}_7$ is

$$\mathbf{H}_7 = \begin{pmatrix} \mathbf{H}_{7_t} \\ \mathbf{H}_{7_b} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{5}$$

Notice that $\mathbf{H}_{N_t}$ computes the pseudo random key sequence, whereas $\mathbf{H}_{N_b}$ the feedback sequence. Therefore, once selected the number N of bits of sequences, we can generate the pseudo random sequences of N bits with the help of the matrix $\mathbf{H}_N$

$$\mathbf{U}_{k+1} = H_N \mathbf{U}_k, \quad k = 1, 2, \ldots \tag{6}$$

where $\mathbf{U}_k = [\mathbf{x}\,\mathbf{y}]^T$ corresponds to the first inputs of function h, and $\mathbf{U}_{k+1}$ is composed by the next inputs of h; note that $\mathbf{U}_{k+1}$ is formed by the generated pseudo random key and the feedback sequence.

4.1. Modified Generator

As it was pointed out in Ref. [3], a generating scheme consisting of three coupled transformations h is proposed to attain an asymptotically unpredictable generator under a random search attack. This proposal is shown in Figure 7, and it is explained briefly. Inside the new generator two copies of the basic transformation h are iterated autonomously from their initial words generating two sequences, $\{p_k\}_{k\geq 0}$ and $\{q_k\}_{k\geq 0}$. The third copy, called the x-map, is iterated in a slightly different manner, the function h in the x-map is driven by the autonomous p-map and q-map according to $x_k = h(p_k, q_k)$. The three maps generate pseudo random sequences, but only the x sequence is released. In order to prevent predictability, the first two words are generated, used and destroyed inside this key generator, therefore they are not available externally. Since the sequences p_k and q_k have a length of N bits each and the required inputs of the h transformation must be one of N bits and the other of $(N+1)$ bits, the missing bit is obtained by applying an addition modulo 2 operation between the two respective LSB's that become the MSB's of their respective previous inputs of the maps. Of course, there exists different manners to generate this missing bit, but we consider this way. The above scheme has just been been proposed a few years ago, but it was only recently implemented by us and studied in terms of the matrix sequence in [15]. With this approach, the new pseudo-random keys are computed as

$$\mathbf{X}_N = \mathbf{H}_{N_t}\mathbf{V}_N \tag{7}$$

where $\mathbf{X}_N = \{x_1, x_2, \ldots, x_N\}^T$, $\mathbf{H}_{N_t}$ is the top matrix of $\mathbf{H}_N$, and $\mathbf{V}_N = \{p_1, \ldots, p_N, q_1, \ldots, q_{N+1}\}^T$. For example, considering $N = 7$, we have

$$\mathbf{X}_7 = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \\ q_7 \\ q_8 \end{pmatrix} = H_{7_t}\mathbf{V}_7, \tag{8}$$

where we calculate p_i and q_i as it was explained above.

4.2. Performance Analysis

Since the PRNG is practically the main part of an encryption system, it has been evaluated with some common statistical tests and for completeness, it is also evaluated by the NIST test suite [19].

We applied the following tests: one-dimensional histogram to observe the distribution of data, the correlation between data, and the Fourier transform to check the spectral properties.

A sequence of one million integers of 15 bits was generated from an unbiased choice of the seed. Figure 8 shows its histogram and its respective profile in the range of $[10, 60]$. We can observe that this statistical property follows a Gaussian distribution type, which we consider as a good result for our purposes.

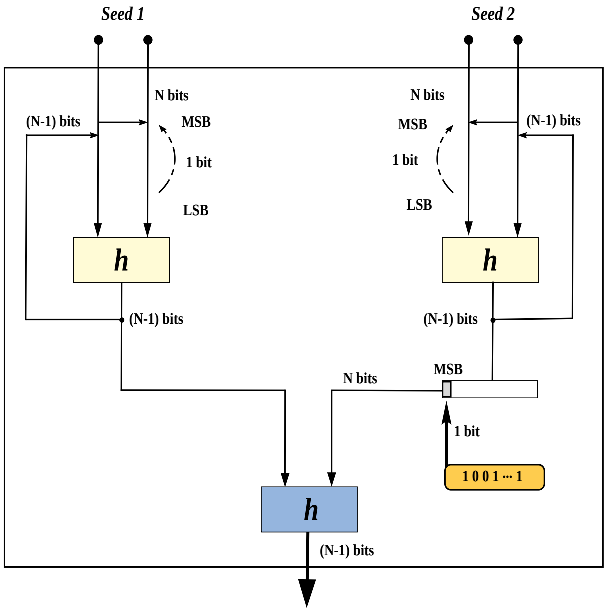


Figure 7. A generating scheme consisting of three coupled transformations.

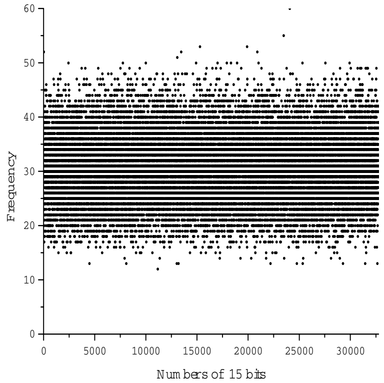


Figure 8. Histogram and its profile of a sequence of 1 million integers of 15 bits.

On the other hand, Figure 9 shows the estimated correlation between successive keys in a sequence. A uniform distribution can be easily observed, showing that there is no correlation between successive keys, suggesting they are random variables.

In terms of frequency, we calculated the Fourier transform of a sequence of 100000 samples. Figure 10 shows the normalized Fourier transform of the considered sequence and its profile. One can see that the data have a spread spectrum, i.e., they are uniformly distributed in all frequencies. The same analysis was carried out for more samples obtaining similar results.

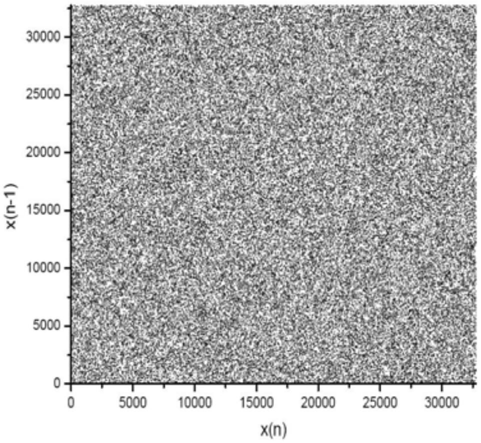


Figure 9. Distribution of the random numbers for n and $n-1$.

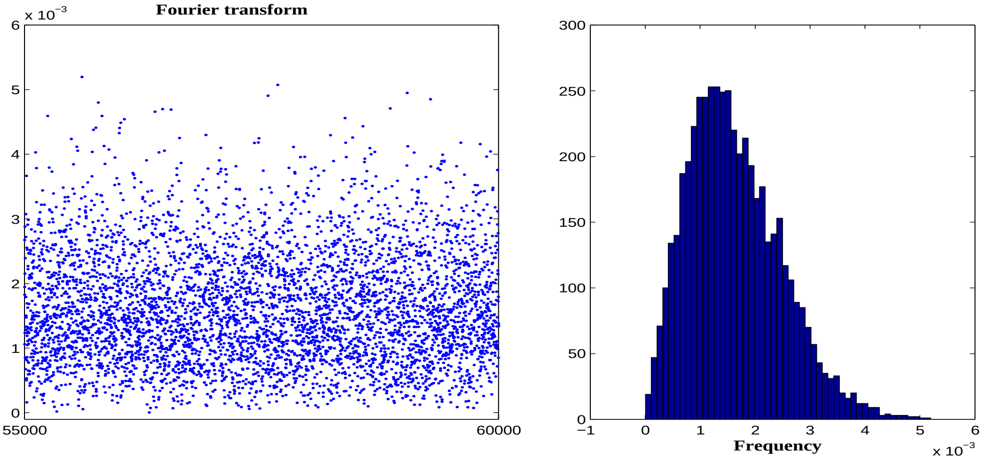


Figure 10. Fourier transform, with its respective profile, of one sample of 5000 data in the interval 55000 – 60000.

With the previous tests, we can conclude that the generated sequences are well-behaved random, the data distribution is uniform, uncorrelated, and its frequency spectrum is extended.

In our study, the PRNG is also evaluated by the NIST suite. The main reason is that this suite has several appealing properties [19, 20]. For instance, it is uniform, it is composed by a number of well known tests and, for all of them, an exhaustive mathematical treatment is available. In addition, the source code of all tests in the suite is publicly available and it is regularly updated [19]. In fact, Reference [19] mentions that the NIST suite may be useful as a first step in determining whether or not a generator is suitable for a particular cryptographic application. The NIST suite is a statistical package consisting of 15 tests that focus on a variety of different types of non-randomness that could exist in a sequence.

These tests are listed in Table 1. Details of the tests can be found in Reference [19].

Table 1. List of NIST Statistical Tests.

Number	Test name
1	The Frequency (Monobit) Test
2	Frequency Test within a Block
3	The Runs Test
4	Tests for the Longest-Run-of-Ones in a Block
5	The Binary Matrix Rank Test
6	The Discrete Fourier Transform (Spectral) Test
7	The Non-overlapping Template Matching Test
8	The Overlapping Template Matching Test
9	Maurer's "Universal Statistical" Test
10	The Linear Complexity Test
11	The Serial Test
12	The Approximate Entropy Test
13	The Cumulative Sums (Cusums) Test
14	The Random Excursions Test
15	The Random Excursions Variant Test

In order to investigate the performance of the generator, we have considered $m = 100$ samples of 10^6 bit sequences, where each sequence, of $N = 15$ bits, has been generated from a randomly chosen seed using one and three transformations.

We have computed the $P-$values corresponding to each sequence for all the tests of NIST suite. As it was pointed out in [19], a $P-$value corresponds to the probability (under the null hypothesis of randomness) that the chosen statistical test will assume values that are equal to or worse than the observed statistical test value when considering the null hypothesis. For the analysis of $P-$values obtained from various statistical tests, we have fixed the significance level at $\alpha = 0.01$, which means that about 1% of the sequences are expected to fail. A sequence passes a statistical test whenever the $P-$value $\geq \alpha$ and fails otherwise. For each statistical test, the proportion of sequences that pass is computed and analyzed accordingly. It is not sufficient to look solely at the acceptance rates and declare that the generator be random if they seem fine. If the test sequences are truly random, the $P-$values calculated are expected to appear uniform in $[0, 1]$. For the interpretation of test results, NIST has adopted two approaches, (1) the examination of the proportion of sequences that pass a statistical test and (2) the distribution of $P-$values to check for uniformity. The respective results are shown in Figures 11-12, and it is observed that by using one transformation, the generator does not pass all tests, see Figure 11 (a), but nevertheless it is uniformly distributed, Figure 11 (b). In fact, it has been observed that this PRNG can generate high-quality random numbers using one or three transformations as the size of keys is increased, i.e., the longer the length of the generated numbers the better is the quality of the random numbers we obtain [15].

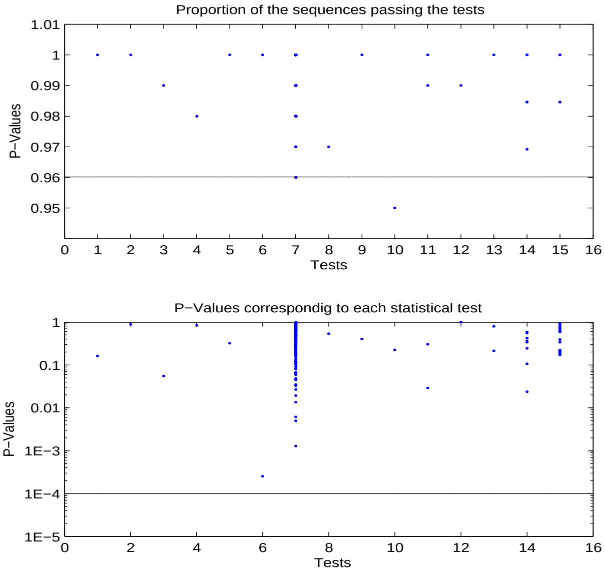


Figure 11. (a) Proportions (TOP), and (b) $P-$values$_T$ (BOTTOM), corresponding to $N = 15$ bits and one transformation. Dashed line separates the success and failure regions.

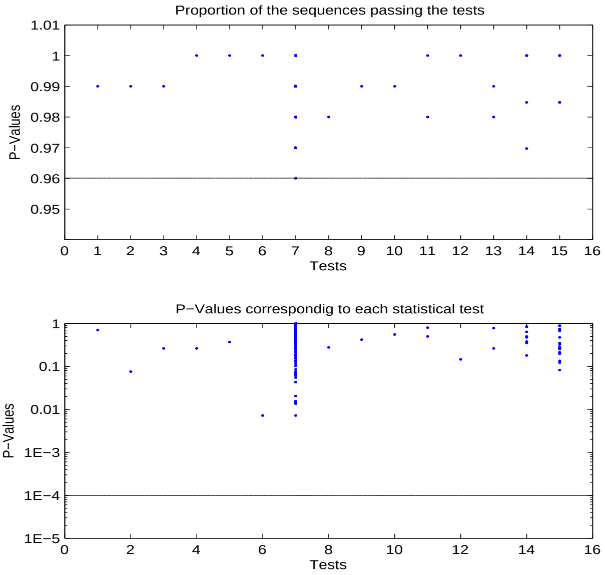


Figure 12. (a) Proportions (TOP), and (b) $P-$values$_T$ (BOTTOM), corresponding to $N = 15$ bits and three transformations. Dashed line separates the success and failure regions.

4.3. Multifractal Properties of the Matrix H$_N$

Since the evolution of the sequence matrix $\mathbf{H}_N$ is based on the evolution of the CA rule 90, the structure of the patterns of bits of the latter are directly reflected in the structure of the entries of $\mathbf{H}_N$. There is recent literature on the multifractal properties of cellular automata for some set of rules, see [11, 12, 13, 14]. In Reference [14], we used the technique of detrended fluctuation analysis based on the discrete wavelet transform (WMF-DFA) to quantify the intrinsic multifractal behavior of the ECAs for rules 90, 105, and 150. Here, in the same spirit as in Reference [14, 15], we analyze the sum of ones in the sequences of the rows of the matrix $\mathbf{H}_N$ with the db-4 wavelet, a wavelet function that belongs to the Daubechies family [16].

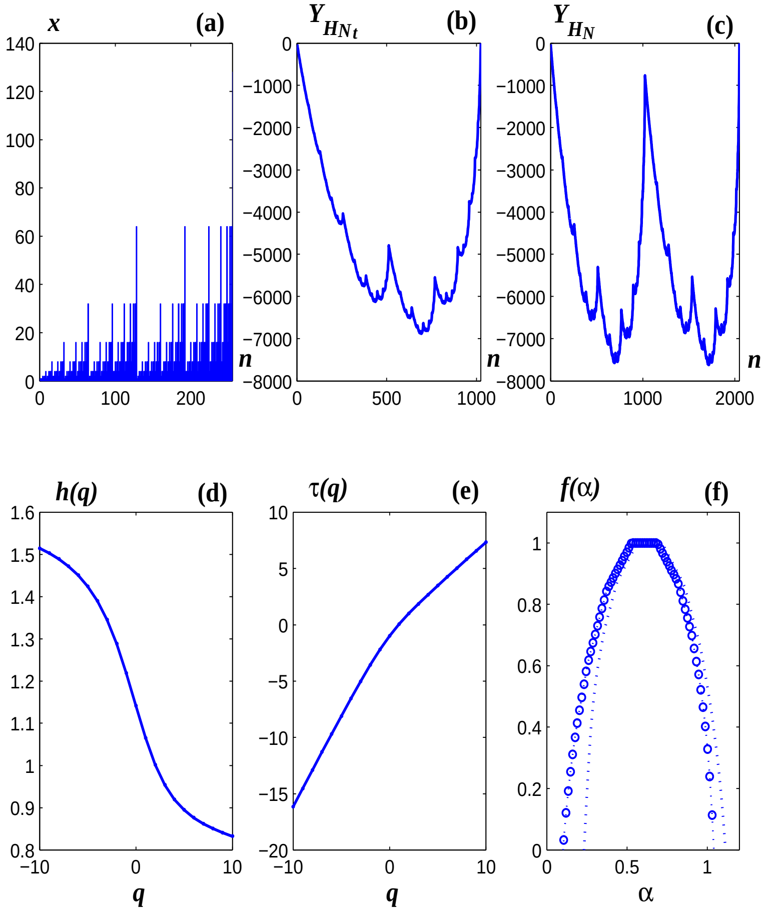


Figure 13. (a) Time series of the row signal of $\mathbf{H}_{1023}$. Only the first 2^8 points are shown of the whole set of $2^{10}-1$ data points. Profiles of the row signal of (b) $\mathbf{H}_{N_t}$ and (c) $\mathbf{H}_N$. (d) Generalized Hurst exponent $h(q)$. (e) The τ exponent, $\tau(q) = qh(q) - 1$. (f) The singularity spectrum $f(\alpha) = q\frac{d\tau(q)}{dq} - \tau(q)$. The calculations of the multifractal quantities h, τ, and $f(\alpha)$ are performed with the wavelet-based WMF-DFA. Dotted points correspond to the row signal of $\mathbf{H}_N$.

The results for one row sum, $\mathbf{H}_{1023}$, is illustrated in Figure 13, and we confirm the multifractality of the time series since we get a τ spectrum with two slopes in both cases. The strength of the multifractality is roughly measured with the width $\Delta\alpha = \alpha_{\max} - \alpha_{\min}$

of the parabolic singularity spectrum $f(\alpha)$ on the α axis. The width $\Delta\alpha_{\mathbf{H}_{1023}} = 1.16 - 0.212 = 0.948$, and the most "frequent" singularity occurs at $\alpha_{\mathrm{mf}_{\mathrm{H}_{1023}}} = 0.694$. We notice that the strongest singularity, $\alpha_{\min}$, and the weakest singularity, $\alpha_{\max}$, are very similar as well as the most "frequent" singularity. These results are in a good agreement with those obtained in Reference [14] for the rule 90, although the spectra of the top matrix present a slight shifting to the right. In fact, this behavior is more evident in the row signals of $\mathbf{H}_N$ (Figure 13(c)), where their corresponding spectra (dotted points) are shown in Figure 13 (f).

5. Wavelet Analysis

5.1. Introduction

Fourier analysis is a well established and suited tool for several kinds of signals, whose statistical properties do not vary with time. The Fourier technique decomposes a signal into harmonic components, where the basis functions are trigonometric. Another alternative to the Fourier approach for analyzing signals is the wavelet transform(WT) [16]. The WT has been introduced and developed to study a large class of phenomena such as image processing, data compression, chaos, fractals, etc. The basis functions of the WT have the key property of localization in time (or space) and in frequency, contrary to what happens with the trigonometric functions. In fact, the WT works as a mathematical microscope on a specific part of a signal to extract local structures and singularities [16]. In addition, an important property of the WT is that it can be turned into sparse expansions, which means that any signal can be quite accurately represented by a small part of the derived coefficients. This property makes WT an effective tool for data compression, situation which interests us.

5.2. Wavelet Transform

Concerning data compression, we consider the orthogonal discrete wavelet transform(DWT). This is only one of the different forms of wavelet transforms [16], by which the wavelets are associated to orthonormal bases of $L^2(\mathbb{R})$.

The representation of a function or process $x(t)$ with the DWT is given in terms of shifted and dilated versions of the wavelet function $\psi(t)$ and its associated scaling function $\varphi(t)$ [16, 17]. Within this framework and considering that the scaling and wavelet functions

$$\varphi_{m,n}(t) = 2^{m/2}\varphi(2^m t - n), \qquad \psi_{m,n}(t) = 2^{m/2}\psi(2^m t - n), \quad m, n \in \mathbb{Z} \tag{9}$$

form an orthonormal basis, one can write the expansion of $x(t)$ as follows

$$x(t) = \sum_n \left(a_{m_0,n}\varphi_{m_0,n}(t) + \sum_{m=m_0}^{M-1} d_{m,n}\psi_{m,n}(t) \right), \tag{10}$$

where the scaling or approximation coefficients $a_{m,n}$ and the wavelet or detail coefficients $d_{m,n}$ are defined as

$$a_{m,n} = \int x(t)\varphi_{m,n}(t)dt, \qquad d_{m,n} = \int x(t)\psi_{m,n}(t)dt, \tag{11}$$

with m and n denoting the dilation and translation indices, respectively.

To calculate $a_{m,n}$ and $d_{m,n}$, Mallat developed the fast wavelet transform (FWT) in which the multiresolution analysis (MRA) approach is involved [16, 17]. The FWT algorithm connects, in an elegant way, wavelets and filter banks, where the multiresolution signal decomposition of a signal X, based on successive decomposition, is represented by a series of approximations and details which become increasingly coarse. At the beginning, the signal is split into two parts, an approximation and a detail part, that together yield the original signal. The subdivision is such that the approximation signal contains the low frequencies, while the detail signal collects the remaining high frequencies. By repeated application of this subdivision rule on the approximation, details of increasingly coarse resolution are separated out, while the approximation itself grows coarser and coarser.

The FWT calculates the scaling and wavelet coefficients at scale m from the scaling coefficients at the next finer scale $m + 1$ using the following formulas

$$a_{m,n} = \sum_k h[k - 2n]a_{m+1,k}, \tag{12}$$

$$d_{m,n} = \sum_k g[k - 2n]a_{m+1,k}, \tag{13}$$

where $h[n]$ and $g[n]$ are typically called low pass and high pass filters in the associated analysis filter bank. In fact, the signals $a_{m,n}$ and $d_{m,n}$ are the convolutions of $a_{m+1,n}$ with the filters $h[n]$ and $g[n]$ followed by a downsampling of factor 2, respectively [16].

Conversely, a reconstruction of the original scaling coefficients $a_{m+1,n}$ can be made from the following combination of the scaling and wavelet coefficients at a coarse scale

$$a_{m+1,n} = \sum_k \left(h[2k - n]a_{m,k} + g[2k - n]d_{m,k}\right) . \tag{14}$$

This corresponds to the synthesis filter bank. This part can be viewed as the discrete convolutions between the upsampled signal $a_{m,l}$ and the filters $h[n]$ and $g[n]$, that is, following an upsampling of factor 2 the convolutions between the upsampled signal and the filters $h[n]$ and $g[n]$ are calculated. The number of levels depends on the length of the signal, i.e., a signal with 2^L values can be decomposed into $(L + 1)$ levels. To initialize the FWT, we consider a discrete time signal $X = \{x[1], x[2], \ldots, x[N]\}$ of length $N = 2^L$. The first application of (12) and (13), beginning with $a_{m+1,n} = x[n]$, defines the first level of the FWT of X. The process goes on, always adopting the $(m + 1)$th scaling coefficients to calculate the "m"th scaling and wavelet coefficients. Iterating (12) and (13) M times, the transformed signal consists of M sets of wavelet coefficients at scales $m = 1, \ldots, M$, and a signal set of scaling coefficients at scale M. There are exactly $2^{(L-m)}$ wavelet coefficients $d_{m,n}$ at each scale m, and $2^{(L-M)}$ scaling coefficients $a_{M,n}$. The maximum number of iterations is $M_{\max} = L$. A three-level decomposition process of the FWT is shown in Figure 14.

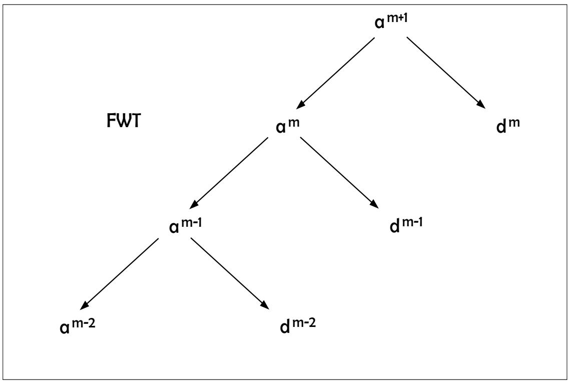


Figure 14. The approximation-detail structure of a three-level FWT.

An important property of the DWT is that the total energy of the signal X may be expressed as follows

$$\mathcal{E}_X = \sum_{n=1}^{N} |x[n]|^2 = \sum_{n=1}^{N} |a_n^M|^2 + \sum_{m=1}^{M} \sum_{n=1}^{N} |d_n^m|^2. \tag{15}$$

This is similar to Parseval's relation in terms of wavelets, where the signal energy can be calculated in terms of the different resolution levels of the corresponding wavelet transformed signal. The property (15) plays a fundamental role in the compression scheme as we will see in the following section.

It is worth to say that the orthonormal Haar wavelet basis was used to perform the FWT, because there is the advantage of implementing a simple algorithm and it is easier to understand. In addition, the algorithm with this wavelet function is memory efficient and reversible without the edge effects that other wavelets present. Other wavelet bases have a slightly higher computational overhead and are conceptually more complex.

The Haar wavelet function is defined as $\psi(t) = \varphi(2t) - \varphi(2t-1)$, where $\varphi(t) = 1$ for all $t \in [0,1]$, and some functions are shown in Figure 15.

The Haar filters associated in the FWT algorithm are $(h[0], h[1]) = (1,1)/\sqrt{2}$ and $(g[0], g[1]) = (1,-1)/\sqrt{2}$.

5.3. Compression Scheme

The complete compression method that we employ herein is shown in Figure 16. First of all, the original signal is segmented into blocks of equal size, and the FWT, with the Haar wavelet, is applied to each block of a length of 2^m samples, with $m < n$. Next, each transformed block is submitted to an elimination process of the transformed coefficients which lie below a threshold value. The key step here is to choose a threshold through an energy criterium. We look at the normalized cumulative energy applied to the ordered transformed coefficients. This is achieved by ordering the magnitude of the transformed coefficients in decreasing order, that is,

$$|y_1| \geq |y_2| \geq \cdots \geq |y_N|$$

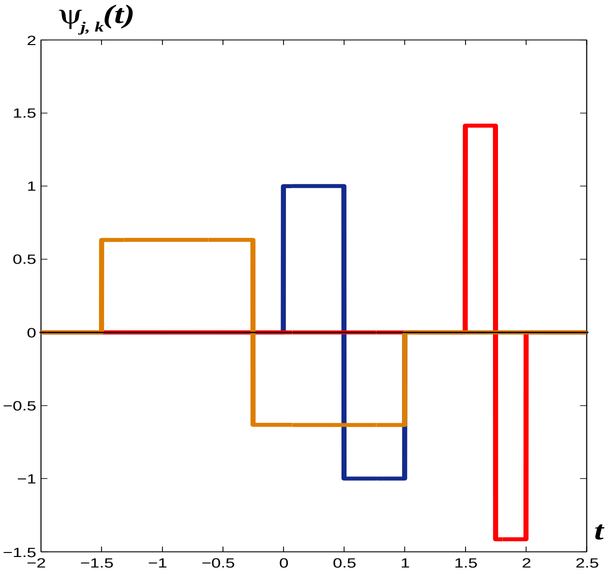


Figure 15. Some functions belonging to the Haar wavelet family.

where $|y_1|$ denotes the largest coefficient of the transformed signal, $|y_2|$ is the next largest, and so forth. The quantity of interest is the normalized cumulative energy defined as

$$\mathcal{E}_{y,n} = \frac{\sum_{k=1}^{n} |y_k|^2}{\sum_{k=1}^{N} |y_k|^2}, \qquad n = 1, \ldots, N, \tag{16}$$

where the denominator is the total energy of the transformed signal Y, $\mathcal{E}_Y$. A graph of $\mathcal{E}_{y,n}$ indicates how many non-zero coefficients are present. The better the representation, the faster the curve will climb towards one. To select the threshold value we consider the magnitude of the coefficient for which the energy percentage in (16) is obtained. With an established threshold value ε, any coefficient in the wavelet transformed data whose magnitude is less than ε will be reset to zero. After thresholding, the non-zero values of the transformed signal are grouped in one vector, which should have a smaller number of coefficients than the original transformed signal. The amount of the obtained compression can be controlled by varying the threshold parameter ε.

Now, we compute a significance map that stores the location information by scanning the thresholded coefficients and outputting a “1” if a significant coefficient is scanned and a “0” if an insignificant coefficient is scanned. The significance map allows us to group the significant coefficients (i.e., nonzero coefficients) in a separate file, and it can be compressed efficiently using, for example, a variable-length code based on run length encoding.

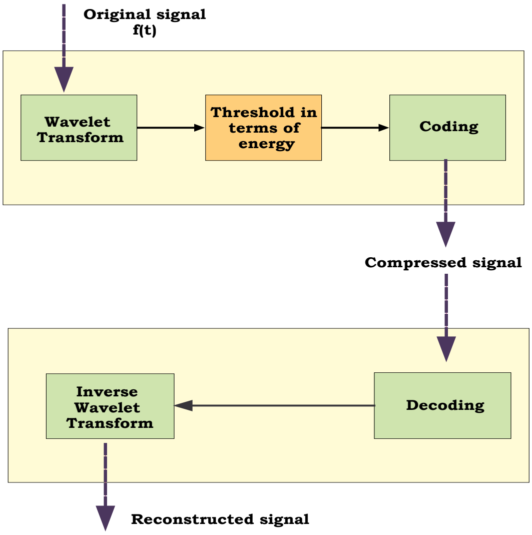


Figure 16. Basic compression scheme based on the wavelet energy.

6. Numerical Implementation

Figure 17 shows a block diagram that depicts the numerical implementation of the CE system. This system was implemented using the graphical programming language of LabVIEW, a trademark of National Instruments, and it consists of two stages. In the first stage, the top block carries out the compression and encryption of information, as described above. Notice that the resulting signal after the compression stage is formed by two signals, the coefficients that survived the threshold, which is the only information that is encrypted, and a binary vector indicating the original positions of the coefficients. The encrypted signal and the binary position vector are available to be transmitted through a channel.

In the second stage, the bottom block performs the reverse process, i.e., the encrypted information received is decrypted, and the decompression stage takes place to the decrypted signal adapted with the binary position vector getting a reconstructed signal.

In such a numerical implementation, we analyzed two different types of speech signals, which will be denoted as $s1$ and $s2$. Both signals were recorded as wav files with a bit resolution of 16 bits and a sampling frequency of 8 kHz and 44.1 kHz, respectively. Thus, a total of $65,536$ samples of $s1$ (about of 8 seconds), and $262,144$ samples of $s2$ (about 6 seconds) were analyzed.

Figure 18 shows the virtual instrument (VI) with the results obtained for the signal $s1$ considering an energy criterium of 90%. In part (a) the signal $s1$ is shown, and we have the transformed signal in part (b). In part (c) is shown the encrypted wavelet coefficients that survive in the compression stage, with a smaller number of samples than the original signal, and finally in part (d) the recovered signal is displayed.

In a similar way, Figure 19 shows the VI with the results obtained for the signal $s2$ considering the same energy criterium of 90%.

Tables 2 and 3 show the compression performance for different energy percentages

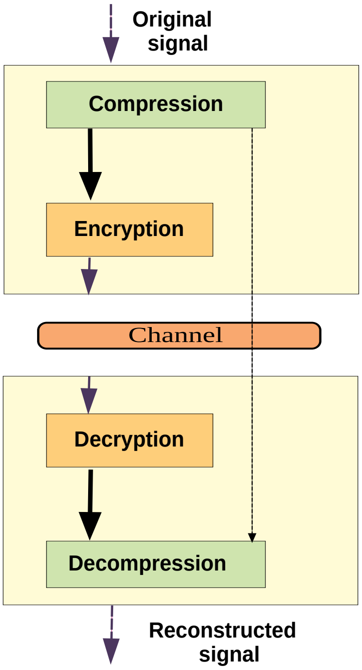


Figure 17. Data compression encryption scheme.

considered in signals $s1$ and $s2$, respectively. Based on these results, we achieved good compression rates for both signals with the Haar wavelet function. When listening to the reconstructed signal, there is not much degradation in the sound quality and the speech is discernible. Of course, we observe that for a higher energy percentage a better sound quality is obtained.

Table 2. Compression rates for signal $s1$.

Percentage of energy	Number of coefficients	Compression ratio
99 %	13913	4.71:1
95 %	5431	12:1
90 %	3231	20.28:1
85 %	2246	29.18:1
80 %	1627	40.28:1

7. Conclusion

In this chapter we have described the implementation of a numerical system that integrates the stages of compression and encryption of voice signals. The compression scheme was

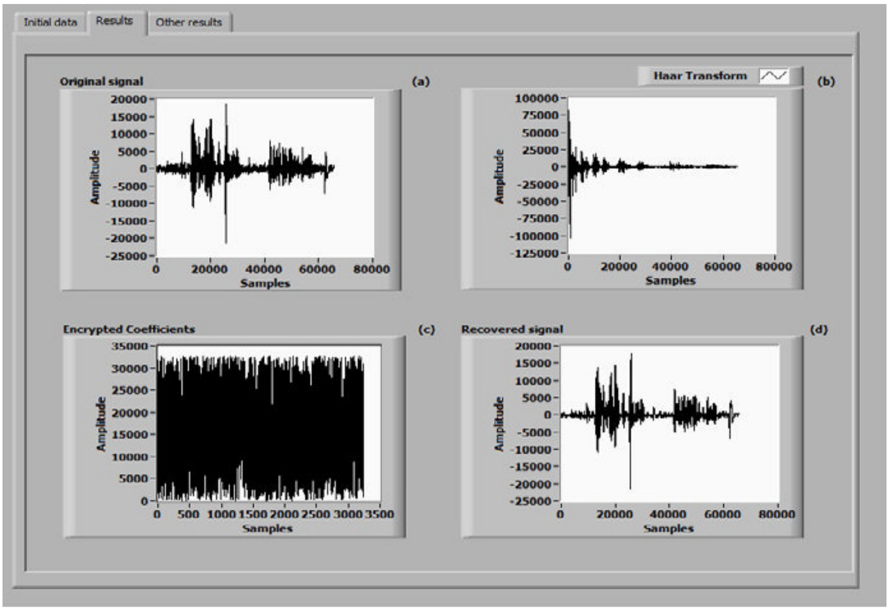


Figure 18. Analysis of the voice signal $s1$ considering an energy criterium of 90%. (a) Original signal $s1$, (b) Haar wavelet transform of the signal $s1$, (c) representation of the encrypted coefficients, and (d) the recovered signal.

Table 3. Compression rates for signal $s2$.

Percentage of energy	Number of coefficients	Compression ratio
99 $\%$	48580	5.4:1
95 $\%$	18769	13.97:1
90 $\%$	10143	25.84:1
85 $\%$	6448	40.65:1
80 $\%$	4458	58.8:1

based on the Haar wavelet transform, while for the encryption process we considered a version of the encryption system proposed in [4]. We have studied, implemented and reviewed the pseudo-random number generator employed in the encryption system, which is based on a rule-90 cellular automaton. This generator in its basic form (using one transformation), and its modified version (with three transformations) is analyzed by means of a sequence matrix $\mathbf{H}_N$. The intrinsic multifractal properties of the sequence matrix in the two versions of the generator are briefly discussed. In addition, the performance of the generated pseudo random sequences is evaluated using some basic tests and the NIST statistical tests. We observe some statistical problems using one transformation, but as was discussed in [15], this PRNG can generate high-quality random numbers using one or three transformations as the size of keys is increased.

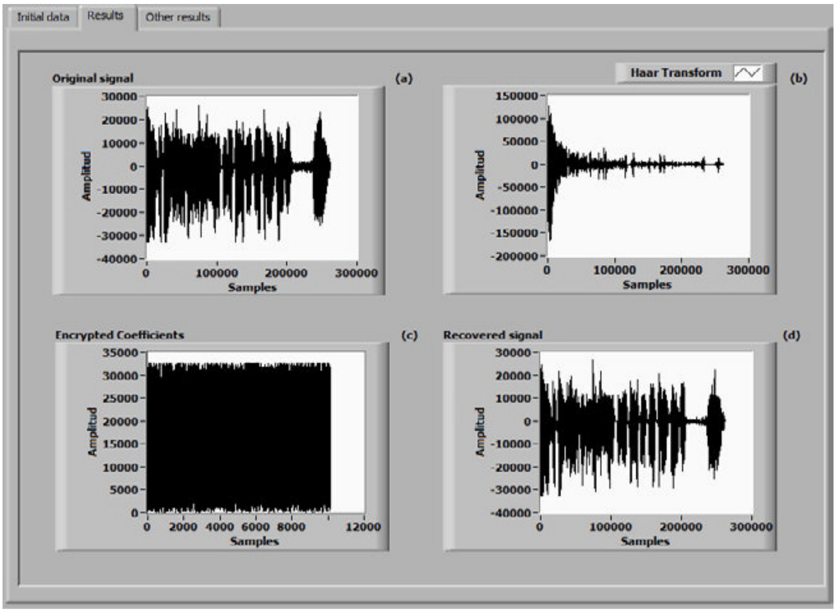


Figure 19. Analysis of the voice signal $s2$ considering an energy criterium of 90%. (a) Original signal $s2$, (b) Haar wavelet transform of the signal $s2$, (c) representation of the encrypted coefficients, and (d) the recovered signal.

On the other hand, the Haar discrete wavelet transform compression gets good rates because a high concentration of energy was presented in a few coefficients of the transformed voice signal. At the same time, the encryption system shows a remarkable security and flexibility to encrypt the information it contains. We believe that this system can be a useful tool in current multimedia applications. In addition, the implemented system is simple, fast, and could be embedded easily in an existing communication system with minimal requirements.

Furthermore, it is believed that the proposed system could have a better performance if the following proposals are carried out or developed: a) optimization in the selection of the threshold, which allows us better compression rates, b) an efficient implementation of the discrete wavelet transform on a FPGA, c) involving the threshold value with the initial seed for the generation of keys.

References

[1] Urías, J.; Salazar, G.; Ugalde, E. *Chaos* 1998, 8, 814-818.

[2] Urías, J.; Ugalde, E.; Salazar, G. *Chaos* 1998, 8, 819-822.

[3] Mejía, M.; Urías, J. *Discrete and Continuous Dynamical Systems* 2001, 7, 115-126.

[4] M. Mejía Carlos,, Ph. D. thesis, Universidad Autónoma de San Luis Potosí, SLP (2001).

[5] Sipper, M.; Tomassini, M. *International Journal of Modern Physics C* 1996, 7, 181-190.

[6] Seredynski, F.; Bouvry, P.; Zomaya, A. Y. *Parallel Computing* 2004, 30, 753-766.

[7] Zied, G.; Mohsen, M.; Medien, Z.; Rached, T. *International Journal of Computer Sciences and Engineering Systems* 2008, 2, 179-185.

[8] Wolfram, S. In *Advances in Cryptology: CRYPTO '85*; Williams, H. C.; Lecture Notes in Computer Science; Springer-Verlag: New York, NY, 1986; Vol. 218, pp 429-432.

[9] Hortensius, P. D.; McLeod, R. D.; Miller, D. M.; Card, H. C. *IEEE Transactions on Computer* 1989, 38, 1466-1473.

[10] Nandi, S.; Kar, B. K.; Chaudhuri, P. P. *IEEE Transactions on Computer* 1994, 43, 1346-1357.

[11] Sanchez, J. R. *International Journal of Modern Physics C* 2003, 14, 491-499.

[12] Sanchez, J. R.; Alonso-Sanz, R. *International Journal of Modern Physics C* 2004, 15, 1461-1470.

[13] Nagler, J.; Claussen, J. C. *Phys. Rev. E* 2005, 71, 067103.

[14] Murguía, J. S.; Perez-Terrazas, J. E.; Rosu, H. C. *Europhysics Letters* 2009, 87, 28003.

[15] Murguía, J. S.; Mejía Carlos, M.; Rosu, H. C.; Flores-Eraña, G. *International Journal of Modern Physics C* 2010, 21, 741-756.

[16] Mallat, S. *A Wavelet Tour of Signal Processing*, 2nd. ed.; Academic Press, San Diego, CA, 1999.

[17] Daubechies, I. *Ten lectures on Wavelets*, SIAM, Philadelphia, PA, 1992.

[18] Patidar, V.; Sud, K. K. *Electronic Journal of Theoretical Physics* 2009, 6, 327-344.

[19] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M.; Vangel, M., Banks, D., Heckert, A., Dray, J, & Vo, S. (2008). *NIST Special Pub. 800-22 Rev. 1*. http://csrc.nist.gov/rng/

[20] Kenny, C. *Random Number Generators: An Evaluation and Comparison of Random.org and Some Commonly Used Generatos*, Trinity College Dublin, Management Sciencie and Information Systems Studies Project Report (2005).

In: Cellular Automata
Editor: Thomas M. Li, pp. 273-288

ISBN 978-1-61761-592-4

Chapter 11

Canonical Factor of Cellular Automata*

Pierre Guillon[†]
Department of Mathematics
University of Turku

Abstract

We study factor subshifts, column factors and the canonical factor of cellular automata in a large setting, that is as endomorphisms of subshifts. Homogeneity of cellular automata makes them share some dynamical properties with their canonical factor; we review some of them.

Introduction

In symbolic spaces, that is spaces of infinite words, it is common to relate topological notions to language notions, with help of the correspondence between the finite words and the cylinders which form a base of the Tychonoff topology. Many properties of topological dynamics can hence be seen when looking at how some patterns of the configuration evolve.

In the case of one-dimensional cellular automata, the homogeneity of the dynamics is such that a pattern of bounded length is enough, in many cases, to derive a property over patterns of any length (which correspond to open sets of arbitrarily small diameter). Interesting results can be obtained by decomposing the evolution of the cells with respect to the evolution of a finite pattern. We give some of them, which consist in classical facts in the theory of cellular automata, and present here a slight generalization to cellular automata defined over SFTs.

In the first section, we give the main definitions of topological and symbolic dynamics. In the second and third sections, we define the fundamental notion of the article, that is the trace, and give some general facts for dynamical systems and cellular automata. In the last three sections, we deal with the three topological notions that are equicontinuity, expansivity and entropy of systems, and present how they are linked to the notion of trace.

*This work has been supported by the Academy of Finland project 131558.

[†] E-mail address: piegui@utu.fi

1. Definitions

Dynamical and symbolic systems. A **dynamical system** is (here) a pair (X, F), where X is a compact metric space and $F : X \to X$ a continuous self-map of X. We shall often omit X when it can be understood from the context. A **subsystem** of (X, F) is its restriction (Y, F) to some closed F-invariant subset $Y \subset X$.

A dynamical system induces an action of the monoid T on X, where $\mathrm{T} = \mathrm{N}$ in general, or $\mathrm{T} = \mathrm{Z}$ if the dynamical system is bijective. In the sequel, we will use T as being implicitly fixed from the definition of the dynamical system (*i.e.* it can stand for either N or Z if we are dealing with a bijective dynamical system, only for N otherwise).

A **morphism** between two dynamical systems (X, F) and (Y, G) is a continuous map $\Phi : X \to Y$ such that $\Phi F = G\Phi$. If it is surjective, it is called a **factor map**, and (Y, G) is a **factor** of (X, F). If it is bijective, it is called a **conjugacy**, and (X, F) and (Y, G) are **conjugate** to one another. If $(X, F) = (Y, G)$, it is called an **endomorphism**, and can be seen itself as a dynamical system. If, besides, it is bijective, it is called an **automorphism**, and can be seen as a bijective dynamical system.

A **symbolic system** is a dynamical system (X, F) where X is totally disconnected. Equivalently, X admits arbitrarily fine clopen partitions (covers of disjoint nonempty closed open sets of any diameter). In that case it is known that, up to homeomorphism, X can be seen as a subset of A^{N}, for some finite alphabet A, endowed with the product topology of the discrete topology. We will restrict our study to symbolic systems of the form (Σ, F), where $\Sigma \subset A^{\mathrm{M}}$ and $\mathrm{M} = \mathrm{Z}$ or $\mathrm{M} = \mathrm{N}$.

A point $x \in A^{\mathrm{M}}$ is called a **configuration**. For $I \subset \mathrm{M}$, we note x_I the restriction of x to I. I can be for example a closed-open interval noted $\mathrm{J}i, j\mathrm{J}$ (or another type of interval, noted similarly), in which case we may allow abuse in the indexes (such as assuming that $x_{\mathrm{J}i,i+k\mathrm{J}} \in A^k$). Note also that, for $k \in \mathrm{N}$, A^k is the same as $A^{\mathrm{J}0,k\mathrm{J}}$. For $k \in \mathrm{N}$, we note $\langle k\rangle = \{i \in \mathrm{M} | \, |i| \leq k\}$, that is $\mathrm{J}{-k}, k\mathrm{K}$ or $\mathrm{J}0, k\mathrm{K}$ whether M is Z or N.

Shifts. We define the **shift** on A^{M} as the particular symbolic system σ defined for $x \in A^{\mathrm{M}}$ and $i \in \mathrm{M}$ by $\sigma(x)_i = x_{i+1}$. It is bijective if $\mathrm{M} = \mathrm{Z}$ (hence we will assume $\mathrm{T} = \mathrm{M}$).

A **subshift** is a subsystem (Σ, σ) of the shift (A^{M}, σ), *i.e.* its restriction to some closed set $\Sigma \subset A^{\mathrm{M}}$ which is invariant by σ^i for any $i \in \mathrm{M}$. For the sake of simplicity, we shall write that Σ is the subshift. The **language over** $I \subset \mathrm{M}$ of Σ is $\mathcal{L}_I(\Sigma) = \{x_I | \, x \in \Sigma\}$. Σ is characterized by its **language** $\mathcal{L}(\Sigma) = \bigcup_{k\in\mathrm{N}} \mathcal{L}_k(\Sigma)$, where $\mathcal{L}_k(\Sigma) = \mathcal{L}_{\mathrm{J}0,k\mathrm{J}}(\Sigma)$, consisting of all the finite patterns that appear in some of its configurations. If $k \in \mathrm{N} \setminus \{0\}$ and $\mathcal{F} \subset A^k$ are such that $\Sigma = \left\{x \in A^{\mathrm{M}} \middle| \, \forall i \in \mathrm{M}, x_{\mathrm{J}i,i+k\mathrm{J}} \notin F\right\}$, then we say that it is an **subshift of finite type** (or SFT) of **order** k.

Fixed a subshift $\Sigma \subset A^{\mathrm{M}}$, the **cylinder** of finite **support** $I \subset \mathrm{M}$ and **central pattern** $u \in A^I$ is the clopen set $[u] = \{x \in \Sigma | \, x_I = u\}$. The **central cylinders** $[u]$, for $u \in A^{\langle k\rangle}$ and $k \in \mathrm{N}$, actually form a base for the topology. When $u \in A^k$ for some $k \in \mathrm{N}$, we can note $[u]_i = \sigma^{-i}[u] = \left\{x \in \Sigma \middle| \, x_{\mathrm{J}i,i+k\mathrm{J}} = u\right\}$.

Cellular automata. A **cellular automaton** over some surjective subshift $\Sigma \subset A^{\mathrm{M}}$ is an endomorphism of it, *i.e.* a dynamical system which commutes with the shift map.

Hedlund's theorem gives a characterization emphasizing the locality aspect of the map.

Theorem 1 ([1]). *Let $\Sigma \subset A^{\mathbb{M}}$ and $\Gamma \subset B^{\mathbb{M}}$ be two subshifts and Φ a morphism from Σ into Γ. Then there exists a finite* **neighborhood** *$I \in \mathbb{M}$ and a* **local rule** *$\phi : \mathcal{L}_I(\Sigma) \to B$ such that for any configuration $x \in \Sigma$ and any cell $i \in \mathbb{M}$, $\Phi(x)_i = \phi(x_{i+I})$.*

Proof. As clopen sets, each of the $|B|$ preimages $\Phi^{-1}([b])$, for $b \in B$, can be decomposed into a finite union of cylinders. Let I be a neighborhood containing all the supports of these cylinders. By construction, $\Phi(x)_0$ depends only on the values of x_I. By shift-invariance, for any $i \in \mathbb{M}$, $\Phi(x)_i = \sigma^i\Phi(x)_0 = \Phi\sigma^i(x)_0$ depends only on the values of $\sigma^i(x)_I$. □

In particular, any cellular automaton F over a subshift $\Sigma \subset A^{\mathbb{M}}$ admits an **anchor** $m \in \mathbb{N}$, an **anticipation** $n \in \mathbb{N}$ and a **local rule** $f : \mathcal{L}_{\llbracket -m,n\rrbracket}(\Sigma) \to A$ such that for any configuration $x \in \Sigma$ and any cell $i \in \mathbb{Z}$, $F(x)_i = f(x_{\llbracket -m,n\rrbracket})$. If $\mathbb{M} = \mathbb{Z}$, we can suppose the anchor and anticipation equal, in which case we call $r = m = n$ **radius** of the cellular automaton. If $\mathbb{M} = \mathbb{N}$, we can suppose the anchor to be 0, in which case we call $r = n$ **radius**.

We will say that Φ is a **letter-to-letter factor map** if we can take a trivial neighborhood $I = \{0\}$.

Inverse limits. The following remark justifies more or less the “factor” terminology.

Remark 1. *If Φ is a factor map of (Y, G) onto (X, F), then (Y, G) is conjugate to the system $(X, F) \otimes_\Phi (Y, G) = (X \otimes_\Phi Y, F \times G)$, where $X \otimes_\Phi Y = \{\, (x, y) \in X \times Y |\ \Phi(y) = x\}$ and $F \times G : (x, y) \mapsto (F(x), G(y))$.*

Furthermore, by a direct induction it can be seen that if $(X_i, F_i)_{0\leq i\leq l}$ is a finite collection of dynamical systems, $l \in \mathbb{N}$ and $(\Phi_i : X_{i+1} \to X_i)_{0\leq i<l}$ a corresponding collection of factor maps, then $\bigotimes_{(\Phi_i)_{0\leq i<l}}(X_i, F_i) = (X_0, F_0) \otimes_{\Phi_0} \ldots \otimes_{\Phi_{l-1}} (X_l, F_l)$ is conjugate to (X_l, F_l). The inverse limit represents some kind of infinite generalization of the $\otimes$ operation.

The **inverse limit** of the sequence $(X_i, F_i)_{i\in\mathbb{N}}$ of dynamical systems with respect to the sequence $(\Phi_i : X_{i+1} \to X_i)_{i\in\mathbb{N}}$ of factor maps is the system $\bigotimes_{(\Phi_i)_{i\in\mathbb{N}}}(X_i, F_i) = (\bigotimes_{(\Phi_i)_{i\in\mathbb{N}}} X_i, \prod_{i\in\mathbb{N}} F_i)$ defined by:

$$\bigotimes_{(\Phi_i)_{i\in\mathbb{N}}} X_i = \left\{ (x_i)_{i\in\mathbb{N}} \in \prod_{i\in\mathbb{N}} X_i \middle| \forall i \in \mathbb{N}, \Phi_i(x_{i+1}) = x_i \right\}$$

$$\prod_{i\in\mathbb{N}} F_i : (x_i)_{i\in\mathbb{N}} \mapsto (F_i(x_i))_{i\in\mathbb{N}} .$$

It basically represents the minimal system which admits all the systems in the sequence as factors, and with the relevant commutations between factor maps.

2. Traces

2.1. Factor Subshifts

Definition 1 (Trace). *Let (X, F) be a symbolic system and $\mathcal{P}$ a clopen partition of X. For any point $x \in X$, there exists a unique clopen set $\mathcal{P}(x) \in \mathcal{P}$ that contains x.*

- *The **trace map** of F relatively to partition $\mathcal{P}$ is defined by:*

$$\begin{array}{rccl} T_F^{\mathcal{P}}: & X & \to & \mathcal{P}^{\mathrm{T}} \\ & x & \mapsto & (\mathcal{P}(F^t(x)))_{t\in\mathrm{T}} \,. \end{array}$$

- *The **trace** of F relatively to $\mathcal{P}$ is its image set $\tau_F^{\mathcal{P}} = T_F^{\mathcal{P}}(X)$.*

If we see $\mathcal{P}^{\mathrm{T}}$ as a symbolic space over alphabet $\mathcal{P}$, then the trace $T_F^{\mathcal{P}}$ is continuous since the preimage $(T_F^{\mathcal{P}})^{-1}([u])$ of any cylinder $[u]$ of $\mathcal{P}^{\mathrm{T}}$ of support $J \subset \mathrm{T}$, with $u \in \mathcal{P}^J$, is a finite intersection $\bigcap_{t\in J} F^{-t}(u_t)$ of open sets. Besides, we easily remark that, for any $x \in X$ and any $t \in \mathrm{T}$:

$$T_F^{\mathcal{P}} F^t(x) = (\mathcal{P}(F^{s+t}(x)))_{s\in\mathrm{T}} = \sigma^t T_F^{\mathcal{P}}(x) \,.$$

Hence the trace $\tau_F^{\mathcal{P}}$ (implicitly endowed with the shift map) is a *factor subshift* of F.

Conversely, any factor map Φ of F onto a subshift $(\Sigma \subset A^{\mathrm{T}}, \sigma)$ is, up to letter renaming, the trace map of F relatively to the partition $\mathcal{P} = \left\{ \Phi^{-1}([a]) \middle| a \in A \right\}$. Hence the factor subshifts of a symbolic system are essentially its traces.

Remark 2. *If $\mathcal{P}$ and $\mathcal{P}'$ are two clopen partitions of a space X such that $\mathcal{P}$ is finer than $\mathcal{P}'$, then we have a decomposition $T_F^{\mathcal{P}'} = \Pi T_F^{\mathcal{P}}$, where $\Pi : \tau_F^{\mathcal{P}} \to \tau_F^{\mathcal{P}'}$ is the letter-to-letter factor map of local rule:*

$$\begin{array}{rccl} \pi : & \mathcal{P} & \to & \mathcal{P}' \\ & U & \mapsto & V \textit{ where } U \subset V \,. \end{array}$$

Example 3. *Let $\Sigma \subset A^{\mathrm{M}}$ be a subshift and $\mathcal{P}$ a clopen partition of Σ that distinguishes the letters, i.e. at least as fine as the partition $\{ [a] | a \in A\}$ into cylinders of width 1. Remark 2 gives us a letter-to-letter factor map $\Phi : T_\sigma^{\mathcal{P}}(\Sigma) \to \Sigma$ whose local rule maps a trace to the central letter of some corresponding configuration, i.e. $\Phi = (T_\sigma^{\mathcal{P}})^{-1}$ is a conjugacy of the trace $T_\sigma^{\mathcal{P}}(\Sigma)$ onto Σ.*

2.2. Generators

A **generator** of a symbolic system F over X is a countable family of clopen partitions $(\mathcal{P}_i)_{i\in\mathrm{N}}$ such that any factor map Ψ of F onto some subshift $\Gamma \subset B^{\mathrm{T}}$ can be written $\Psi = \Psi' \tau^{\mathcal{P}_i}$, for some $i \in \mathrm{N}$ and some factor map Ψ'. It is a **letter-to-letter generator** if, besides, Ψ' can be taken a letter-to-letter factor map.

This dynamical notion of generator is linked to the topological notion of base, since the traces relatively to some base of partitions form a generator.

Proposition 1. *Any base of clopen partitions is a letter-to-letter generator.*

Proof. Let $(p_i)_{i\in\mathrm{N}}$ a base of clopen partitions and Ψ a factor map onto some subshift. By previous remarks, we can assume that $\Psi = T_F^{\mathcal{P}}$ for some clopen partition $\mathcal{P}$ of X. Let $i \in \mathrm{N}$ be some index of a partition $\mathcal{P}_i$ finer than $\mathcal{P}$. By Remark 2, there is a letter-to-letter factor map $\Phi : \tau_F^{\mathcal{P}_i} \to \tau_F^{\mathcal{P}}$ such that $T_F^{\mathcal{P}} = \Phi T_F^{\mathcal{P}_i}$. □

We can also link the previously-defined notion of generator with that of inverse limit.

Proposition 2. *If F is the inverse limit of a sequence $(\Sigma_i)_{i\in\mathrm{N}}$ of subshifts, then it admits a generator $(\mathcal{P}_i)_{i\in\mathrm{N}}$ whose trace $\tau_F^{\mathcal{P}_i}$ is conjugate to Σ_i for any $i \in \mathrm{N}$.*

Proof. By definition of the product topology, X admits the family $(\mathcal{P}_i)_{i\in\mathrm{N}}$ as a base of partitions, where $\mathcal{P}_i$ is defined as the preimage by the projection $\pi_{\mathrm{J}0,i\mathrm{K}}$ of the partition into cylinders of width $i+1$ of the subshift $\Sigma_0 \otimes \ldots \otimes \Sigma_i$:

$$\mathcal{P}_i = \left\{ \pi^{-1}_{\mathrm{J}0,i\mathrm{K}}([u]) \middle| u \in \mathcal{L}_{i+1}(\Sigma_0 \otimes \ldots \otimes \Sigma_i) \right\} .$$

Since this projection $\pi_{\mathrm{J}0,i\mathrm{K}}$ does not depend on other systems of the base, we can see that the trace $\tau_F^{\mathcal{P}_i}$ is conjugate to the trace $T_\sigma^{\langle i\rangle}(\Sigma_0 \otimes \ldots \otimes \Sigma_i)$, itself conjugate to $\Sigma_0 \otimes \ldots \otimes \Sigma_i$ by Example 3, and hence to Σ_i by Remark 1. □

The converse is true as soon as we order the traces from the finest to the coarsest.

Proposition 3. *If $(\mathcal{P}_i)_{i\in\mathrm{N}}$ is a fineness-increasing sequence of partitions of X, then any symbolic system (X, F) is essentially the inverse limit (Y, G) of the family of traces $(\tau_F^{\mathcal{P}_i})_{i\in\mathrm{N}}$ (relatively to the canonical projections).*

Proof. The morphism:

$$\begin{array}{cccc} \Phi: & X & \to & Y \\ & x & \mapsto & (T_F^{\mathcal{P}_i}(x))_{i\in\mathrm{N}} \end{array}$$

is bijective since $\Phi^{-1}((y_i)_{i\in\mathrm{N}}) = \bigcap_{i\in\mathrm{N}}(T_F^{\mathcal{P}_i})^{-1}(y_i)$ is an intersection of closed sets, that is decreasing if and only if $y \in Y$, and whose diameter converges towards 0. □

Corollary 4. *For any generator $(\mathcal{P}_i)_{i\in\mathrm{N}}$ of a symbolic system (X, F), there is an increasing sequence $(k_i)_{i\in\mathrm{N}}$ of integers such that (X, F) is the inverse limit of $(\tau_F^{\mathcal{P}_{k_i}})_{i\in\mathrm{N}}$.*

Proof. Let $\mathcal{Q}_i$ the partition into balls of radius 2^{-i}. Let us build, by recurrence on $i \in \mathrm{N}$, indexes k_i and factor maps of $\tau_F^{\mathcal{P}_{k_{i+1}}}$ onto $\tau_F^{\mathcal{P}_{k_i}}$. Initially, we can take $k_0 = 0$. Suppose now that $i \in \mathrm{N}$ and k_i are already built. By Proposition 1, there is a decomposition $T_F^{\mathcal{P}_{k_i}} = \Psi_i T_F^{\mathcal{Q}_{r_i}}$, where $r_i \in \mathrm{N}$ and Ψ_i is a letter-to-letter factor map of $\tau_F^{\mathcal{Q}_{r_i}}$ onto $\tau_F^{\mathcal{P}_{k_i}}$. We can suppose without loss of generality that $r_i \geq i$ (otherwise take the maximum with i and compose with a projection). By hypothesis, there exist an index $k_{i+1} \in \mathrm{N}$ and a factor map Ψ'_i of $\tau_F^{\mathcal{P}_{k_{i+1}}}$ onto $\tau_F^{\mathcal{P}_{r_i}}$, such that $T_F^{\mathcal{Q}_{r_i}} = \Psi'_i T_F^{\mathcal{P}_{k_{i+1}}}$. Hence, we have $T_F^{\mathcal{P}_{k_i}} = \Psi_i \Psi'_i T_F^{\mathcal{P}_{k_{i+1}}}$. Remark that the product $\prod_{i\in\mathrm{N}} \Psi_i$ is a conjugacy of the inverse limit of the sequence $(\tau_F^{\mathcal{Q}_{r_i}})_{i\in\mathrm{N}}$ onto the inverse limit of the sequence $(\tau_F^{\mathcal{P}_{k_i}})_{i\in\mathrm{N}}$, of inverse conjugacy $\sigma \prod_{i\in\mathrm{N}} \Psi'_i$. By Proposition 3, (X, F) is conjugate to the former, hence to the latter. □

2.3. Column Factors

If F is a symbolic system over $\Sigma \subset A^{\mathrm{M}}$, a visually intuitive candidate as a generator is the canonical base consisting of the cylinders.

Definition 2 (Column factor).

- *The* ***column factor*** *over the finite support $I \subset \mathbb{M}$ is the trace $\tau_F^I = T_F^I(\Sigma)$, where:*

$$\begin{array}{rccl} T_F^I: & \Sigma & \to & (A^I)^{\mathbb{T}} \\ & x & \mapsto & (F^t(x)_I)_{t\in\mathbb{T}}\ . \end{array}$$

- *The* ***central column factor*** *of radius $r \in \mathbb{N}$ is the trace $\tau_F^{\langle r\rangle}$.*

A central column factor corresponds to an observation of the system evolution through some finite window. The central cylinders being a base of partitions, we can see, as implicitly stated in [2], that the family of central column factors is a letter-to-letter generator and the symbolic system is an inverse limit of it; hence we can essentially restrict our study of factor subshifts to column factors.

3. Traces of Cellular Automata

The particular case of cellular automata allows shift-invariance to link any column factor to some central column factor. Formally, if F is a cellular automaton over some surjective subshift $\Sigma \subset A^{\mathbb{M}}$, then for any cell $i \in \mathbb{M}$ and any finite support $I \subset \mathbb{M}$, $\tau_F^{i+I} = T_F^I(\sigma^i(\Sigma)) = T_F^I(\Sigma) = \tau_F^I$, up to reindexing of the letters in the words.

Canonical factor. The canonical factor was first defined for onesided cellular automata over full shifts in [3]. It corresponds to a trace whose width is equal to the radius of the cellular automaton, *i.e.* the minimal width that cannot be overpassed by information, since if we cut by this width the configuration into a right and a left part, a cell of the left part of the configuration cannot see, in its neighborhood, any of the cells in the right part. Note that our definition of anchor and anticipation, contrary to some other versions, forces the neighborhood of a cell to contain the cell itself.

Definition 3 (Canonical factor). *Let F be a cellular automaton of anchor $m \in \mathbb{N}$ and anticipation $n \in \mathbb{N}$ on some SFT Σ of order $(m+n)$. The* ***left canonical factor*** *of F is the trace τ_F^m whose width is the anchor m. Similarly, its* ***right canonical factor*** *is τ_F^n. Its* ***canonical factor*** *is the widest between the two, i.e. τ_F^r, where $r = \max(m,n)$ is the radius.*

Note that this definition is actually relevant for any cellular automaton over any surjective SFT; if the anchor is more than the order, then it can be increased, and vice-versa.

Overlap.

Definition 4 (Overlap).

- *Let I an interval of $\mathbb{M}$, J an interval of $\mathbb{T}$ and $z = (z^j)_{j\in J}$ a (finite or infinite) word on alphabet A^I, where $z^j = (z_i^j)_{i\in I}$ for any $j \in J$. For any $I' \subset I$, we write the projection $\pi_{I'}(z) = ((z_i^j)_{i\in I'})_{j\in J}$.*

- *Let $k \in \mathrm{N} \setminus \{0\}$, J an interval of T and $z = (z^j)_{j \in J}, w = (w^j)_{j \in J}$ two (finite or infinite) words on alphabet A^k. If $\pi_{\mathrm{J}1,k\mathrm{J}}(z) = \pi_{\mathrm{J}0,k-1\mathrm{J}}(w)$, then we say that these words are* ***overlapping*** *and we define their* ***overlap*** *$z \odot w \in A^{k+1}$ such that for any $i \in \mathrm{J}0, k\mathrm{K}\,, j \in J$, $(z \odot w)^j_i = z^j_i$ if $i < k$ and $(z \odot w)^j_i = w^j_{k-1}$ if $i = k$.*

- *If Σ and Γ are two subsets of $(A^k)^J$, we define their* ***overlap*** *$\Sigma \odot \Gamma = \left\{ z \odot w \middle| z \in \Sigma, w \in \Gamma \text{ and } \pi_{\mathrm{J}1,k\mathrm{J}}(z) = \pi_{\mathrm{J}0,k-1\mathrm{J}}(w) \right\}$.*

- *If $l \in \mathrm{N}$, we define the l-**overlap** of $\Sigma \subset (A^k)^J$ by recurrence: $\Sigma^{[0]} = \Sigma$ and for $l \geq 0$, $\Sigma^{[l+1]} = \Sigma^{[l]} \odot \Sigma^{[l]}$.*

- *We say that $\Sigma \subset (A^k)^J$ is* ***self-overlapping*** *if it is equal to the projections $\pi_{\mathrm{J}i,i+k\mathrm{J}}(\Sigma^{[1]})$ of its 1-overlap for $0 \leq i \leq 1$.*

The overlap operation is associative, which allows easy manipulation. In particular, we can note that if $l, l' \in \mathrm{N}$ and $\Sigma \subset (A^k)^J$, then $(\Sigma^{[l]})^{[l']} = \Sigma^{[l+l']}$.

Of course, the overlap operation is increasing: if $\Sigma \subset \Gamma$, then for any $l \in \mathrm{N}$, $\Sigma^{[l]} \subset \Gamma^{[l]}$. We can also make the following remark.

Remark 4. *If $k \in \mathrm{N} \setminus \{0\}$, J an interval of M and $\Sigma \subset (A^k)^J$, then $\Sigma^{[l]}$ is the biggest set $\Gamma \subset (A^{k+l})^J$ such that for any $i \in \mathrm{J}0, l\mathrm{K}$, $\pi_{\mathrm{J}i,i+k\mathrm{J}}(\Gamma) \subset \Sigma$, i.e. :*

$$\Sigma^{[l]} = \left\{ u \in (A^{k+l-1})^J \middle| \forall i \in \mathrm{J}0, l\mathrm{K}\,, \pi_{\mathrm{J}i,i+k\mathrm{J}}(u) \in \Sigma \right\} .$$

Let us now see that the self-overlapping property is preserved by the overlap operation.

Lemma 1. *If $k \in \mathrm{N} \setminus \{0\}$, J an interval of T and $\Sigma \subset (A^k)^J$ a self-overlapping set, then its 1-overlap $\Sigma^{[1]}$ is also self-overlapping.*

Proof. Let us show that $\pi_{\mathrm{J}0,k\mathrm{K}}(\Sigma^{[2]}) = \pi_{\mathrm{J}1,k+1\mathrm{K}}(\Sigma^{[2]}) = \Sigma^{[1]}$. By definition, we already have the inclusion.
Conversely, let $z \in \Sigma^{[1]}$; let us show that $z \in \pi_{\mathrm{J}0,k\mathrm{K}}(\Sigma^{[2]})$ (the other projection can be obtained symmetrically). Note that, by definition, $\pi_{\mathrm{J}1,k\mathrm{K}}(z)$ is a word of Σ, hence by hypothesis of $\pi_{\mathrm{J}0,k\mathrm{J}}(\Sigma^{[1]})$, there exists $z' \in \Sigma^{[1]}$ such that $\pi_{\mathrm{J}0,k\mathrm{J}}(z') = \pi_{\mathrm{J}1,k\mathrm{K}}(z)$. Hence $z \odot z' \in (\Sigma^{[1]})^{[1]}$; z is a projection of $\Sigma^{[2]}$. □

Proposition 5. *Let $k \in \mathrm{N} \setminus \{0\}$, J an interval of T and $l \in \mathrm{N}$. If $\Sigma \subset (A^k)^J$ is self-overlapping, then it is equal to the projections $\pi_{\mathrm{J}i,i+k\mathrm{J}}(\Sigma^{[l]})$ of its l-overlap, for $0 \leq i \leq l$.*

Proof. Let us show this property by recurrence on $l \in \mathrm{N}$. The case $l = 0$ is trivial. Let $l \geq 0$, $k \in \mathrm{N} \setminus \{0\}$ and $\Sigma \subset A^k$ which is self-overlapping; let us show that $\Sigma = \pi_{\mathrm{J}i,i+k\mathrm{J}}(\Sigma^{[l+1]})$ for $0 \leq i \leq l+1$. By Lemma 1, we know that $\Sigma^{[1]}$ is self-overlapping; thus we can apply the recurrence hypothesis: $\Sigma^{[1]} = \pi_{\mathrm{J}i,i+k+1\mathrm{J}}(\Sigma^{[l+1]})$, for $0 \leq i \leq l+1$. But since Σ equals any projection of width k of $\Sigma^{[1]}$, it also equals any projection of $\Sigma^{[l+1]}$. □

For two configurations $x, y \in A^{\mathrm{M}}$ and a cell $i \in \mathrm{M}$, we define the **joint** $x \oplus_i y$ as the configuration z such that $z_k = x_k$ if $k < i$ and $z_k = y_k$ if $k \geq i$. We say that x and y are k-**overlapping** in cell $i \in \mathrm{M}$, where $k \in \mathrm{N}$, if $x_{\mathrm{J}i,i+k\mathrm{J}} = y_{\mathrm{J}i,i+k\mathrm{J}}$. In that case, we can see that it is also the case of x and $x \oplus_i y$, or yet of $x \oplus_i y$ and y.

Traces and overlapping. The following proposition formalizes the fact that, if we impose some trace of width $m+n$, the evolution of some configuration can be decomposed into a left and a right part in the sense that cells of each side do not see the other one.

Proposition 6. *If (Σ, F) is a cellular automaton of anchor $m \in \mathrm{N}$ and anticipation $n \in \mathrm{N}$ on a SFT of order $(m+n)$, J an interval of T, $i \in \mathrm{M}$ and $x, y \in \Sigma$ two configurations such that $T_F^{\mathrm{J}i,i+m+n\mathrm{J}}(x)_J = T_F^{\mathrm{J}i,i+m+n\mathrm{J}}(y)_J$. Then for any step $t \in J$, $F^j(x \oplus_i y) = F^j(x) \oplus_i F^j(y)$.*

Proof. We can see by recurrence on $j \in \mathrm{N} \cap J$ that the neighborhood $F^j(x \oplus_i y)_{\mathrm{J}k-m,k+n\mathrm{J}}$ of any cell $k \in \mathrm{M}$ corresponds to the neighborhood $F^j(x)_{\mathrm{J}k-m,k+n\mathrm{J}}$ if $k < m$, $F^j(y)_{\mathrm{J}k-m,k+n\mathrm{J}}$ otherwise, hence the application of the rule is unaltered. If $j \in J \setminus \mathrm{N}$, we can apply the map F^j to what was obtained in the previous point. □

We can also write a variant of the previous proposition that distinguishes left and right.

Proposition 7. *Let (Σ, F) a cellular automaton of anchor $m \in \mathrm{N}$ and anchor $n \in \mathrm{N}$, $k \in \mathrm{N}$, J an interval of T and x, y two configurations that coincide on some finite segment $x_{\mathrm{J}0,k\mathrm{J}} = y_{\mathrm{J}0,k\mathrm{J}}$ and on the extremity right and left canonical factors $T_F^{\mathrm{J}-m,0\mathrm{J}}(x)_{\mathrm{J}0,J\mathrm{J}} = T_F^{\mathrm{J}-m,0\mathrm{J}}(y)_J$ and $T_F^{\mathrm{J}k,k+n\mathrm{J}}(x)_J = T_F^{\mathrm{J}k,k+n\mathrm{J}}(y)_J$. Then the traces corresponding to the whole segment will coincide: $T_F^{\mathrm{J}-m,k+n\mathrm{J}}(x)_J = T_F^{\mathrm{J}-m,k+n\mathrm{J}}(y)_J$.*

Proof. First, if $x_{\mathrm{J}0,\infty\mathrm{J}} = y_{\mathrm{J}0,\infty\mathrm{J}}$ and $T_F^{\mathrm{J}-m,0\mathrm{J}}(x)_J = T_F^{\mathrm{J}-m,0\mathrm{J}}(y)_J$, then a direct recurrence on $j \in \mathrm{N} \cap J$ allows to see that $F^j(x)_{\mathrm{J}-m,\infty\mathrm{J}} = F^j(y)_{\mathrm{J}-m,\infty\mathrm{J}}$. If $j \in J \setminus \mathrm{N}$, we can apply the map F^j to what was obtained in the previous point. The same can be done on the right side with $T_F^{\mathrm{J}k,k+n\mathrm{J}}$. Combining these two points, we get the expected result. □

Proposition 8. *If (Σ, F) is a cellular automaton and $k \in \mathrm{N}$, then τ_F^k is self-overlapping and for any $l \in \mathrm{N}$, $(\tau_F^k)^{[l]} \subset \tau_F^{k+l}$.*

Proof. Let $x \in \Sigma$ be a configuration. Then for any cell $i \in \mathrm{J}0, l\mathrm{K}$, $\pi_{\mathrm{J}i,i+k\mathrm{J}} T_F^{k+l}(x) = T_F^{\mathrm{J}i,i+k\mathrm{J}}(x)$, hence $T_F^{k+l}(x)$ is in the overlapping $(\tau_F^k)^{[l]}$. Conversely, for any cell $i \in \mathrm{J}0, l\mathrm{K}$, $\pi_{\mathrm{J}i,i+k\mathrm{J}}(\tau_F^{k+l}) = \tau_F^k$, hence τ_F^k is overlapping. □

As a result, overlapping of sufficiently wide column factors are still column factors.

Proposition 9. *Let F be a cellular automaton of anchor $m \in \mathrm{N}$ and anticipation $n \in \mathrm{N}$ on some SFT Σ of order $(m+n)$, $l \in \mathrm{N}$ and $k > m+n$. Then $(\tau_F^k)^{[l]} = \tau_F^{k+l}$.*

Proof. From Proposition 8, we just have to show that the overlapping $(\tau_F^k)^{[l]}$ is included in the trace τ_F^{k+l}, which can be done by recurrence on $l \in \mathrm{N}$. The case $l = 0$ is trivial. Now assume that it is true for some $l \in \mathrm{N}$. Let z be a word of $(\tau_F^k)^{[l+1]}$, which is equal by recurrence hypothesis to $(\tau_F^{k+l})^{[1]}$. Then there exist two configurations x, y such that $T_F^{k+l}(x) = \pi_{\mathrm{J}0,k+l\mathrm{J}}(z)$ and $T_F^{\mathrm{J}1,k+l\mathrm{K}}(y) = \pi_{\mathrm{J}1,k+l\mathrm{K}}(z)$.
By hypothesis, $k+l-1 \geq m+n$, hence Proposition 6 gives $T_F^{k+l+1}(x \oplus_1 y) = z$. □

Combining Propositions 9 and 3 allows to rebuild the cellular automaton from its sufficiently wide traces.

Corollary 10. *A cellular automaton F of anchor $m \in \mathrm{N}$ and anticipation $n \in \mathrm{N}$ is essentially the inverse limit of the overlaps $((\tau_F^{m+n+1})^{[k]})_{k\in\mathrm{N}}$ of the trace whose width is the diameter $m+n+1$ of the neighborhood.*

Compatibility.

Definition 5 (Compatible subshift).

- *A word (or configuration) $(z_j)_{j\in J} \subset (A^k)^J$, with $k \in \mathrm{N}$ and $J \subset \mathrm{T}$, is* ***compatible*** *with the cellular automaton (Σ, F) of anchor $m \in \mathrm{N}$, anticipation $n \in \mathrm{N}$, and local rule $f : \mathcal{L}_{m+n+1}(\Sigma) \to A$ if for any step $j \in J$, $z^j \in \mathcal{L}_k(\Sigma)$ and if $j+1 \in J$ then for any cell $i \in \mathrm{J}m, k-n\mathrm{J}$, $z_i^{j+1} = f(z^j_{\mathrm{J}i-m,i+n\mathrm{J}})$.*
- *A subshift $\Sigma \subset (A^k)^{\mathrm{M}}$ is* ***compatible*** *with (Σ, F) if all of its configurations are.*

Note that this definition is really relevant when k is at least equal to the diameter of F. In that case, Σ represents a candidate for being the trace of width k, in the sense that all the central cells respect the local rule of F.

Remark 5. *Any column factor τ_F^k of width $k \in \mathrm{N}^*$ of a cellular automaton F is compatible with F.*

Conversely, it can be shown that any compatible self-overlapping subshift is a "column subfactor".

Proposition 11. *If $k \in \mathrm{N}$ and $\Sigma \subset (A^k)^{\mathrm{M}}$ is a subshift which is self-overlapping and compatible with some cellular automaton F over some SFT of order $m+n$, with anchor $m \in \mathrm{N}$, anticipation $n \in \mathrm{N}$, then Σ is included in the trace τ_F^{m+n}.*

Proof. Suppose $\mathrm{M} = \mathrm{Z}$ (the construction is similar when $\mathrm{M} = \mathrm{N}$) and $z \in \Sigma$. Being self-overlapping, we can inductively build a sequence $(z^l)_{l\in\mathrm{N}}$ of words, with $z^0 = z$ and for any $l \in \mathrm{N}$, $z^l \in \Sigma^{[2l]}$ and $z^l = \pi_{\mathrm{J}1,k+2l\mathrm{K}}(z^{l+1})$. For $j \in \mathrm{N}$, we define x^j as the unique element of the decreasing intersection $\bigcap_{l\in\mathrm{N}}[z_j^l]_{-l}$. The compatibility and a direct recurrence give that for any cell $i \in \mathrm{Z}$, $x_i^{j+1} = f(x^j_{\mathrm{J}i-m,i+n\mathrm{J}})$, where f is the local rule of F. In particular, we can then see that $\tau_F^k(x^0) = z$. □

The previous result is still true if we take wider m and n, since they can then still be considered as the anchor and anticipation of the same cellular automaton, and their sum can still be considered as the order of the same subshift.

4. Equicontinuity

Let (X, F) be a dynamical system, $\varepsilon \in \mathrm{R}_+ \setminus \{0\}$. A point $x \in X$ is said ε-**unstable** if for any radius $\delta > 0$, there is a point $y \in \mathcal{B}_\delta(x)$ and a step $t \in \mathrm{T}$ for which $\mathrm{d}(F^t(x), F^t(y)) > \varepsilon$. Otherwise the point is said ε-**stable**. A point which is ε-stable for any $\varepsilon > 0$ is said **equicontinuous**.

A dynamical system F is said ε-**sensitive** if all of its points are ε-unstable for some $\varepsilon > 0$. It is said **almost equicontinuous** if its set of equicontinuous points is a residual.

It is **equicontinuous** if for any radius $\varepsilon > 0$, there exists a radius $\delta > 0$ such that for all points $x, y \in X$ with $\mathrm{d}(x, y) < \delta$ and all steps $t \in \mathrm{T}$ we have $\mathrm{d}(F^t(x), F^t(y)) < \varepsilon$. Due to the compactness of the underlying space, it is possible to invert the two quantifiers in the definition of equicontinuity: a dynamical system F is equicontinuous if and only if all of its points are. Moreover, in the case of bijective systems, it is known that the definitions with $\mathrm{T} = \mathrm{Z}$ and $\mathrm{T} = \mathrm{N}$ coincide.

Blocking words. The topological notion of equicontinuity can, in a one-dimensional space, be expressed symbolically in terms of blocking words, which prevent the information transmission.

A word $w \in A^*$ is (i, k)-**blocking** (or simply k-**blocking**), with $i \in \mathrm{M}, k \in \mathrm{N}$, for the symbolic system (Σ, F) if $\forall x, y \in [w]_i, \forall t \in \mathrm{T}, F^t(x)_{\mathrm{J}0,k\mathrm{J}} = F^t(y)_{\mathrm{J}0,k\mathrm{J}}$. Except in trivial cases, we will have $i + k \leq |w|$.

Of course, a word is k-blocking whenever one of its subwords is. Moreover, a k-blocking word is l-blocking for any $l \leq k$.

From the definition, we can see a strong link between word blockingness and point stability.

Remark 6. *Let (Σ, F) be a symbolic system and $k \in \mathrm{N}$. Then the following are true.*

- *A configuration $x \in \Sigma$ is 2^{-k}-stable if and only if $x_{\langle l \rangle}$ is k-blocking for some $l \in \mathrm{N}$.*
- *F is 2^{-k}-sensitive if and only if it admits no k-blocking word.*
- *A configuration is equicontinuous if and only if it admits k-blocking central patterns for any $k \in \mathrm{N}$.*

Blocking words are particularly relevant in cellular automata, in which case a particular blocking width is enough to have any. More formally, if (Σ, F) is a cellular automaton of anchor m and anticipation n, w an (i, m)-blocking (resp. (i, n)-blocking) word and $x \in [w]_i$, then for any configuration $y \in \Sigma$ such that $y_{\mathrm{J}i,\infty\mathrm{J}} = x_{\mathrm{J}i,\infty\mathrm{J}}$ (resp. $y_{\mathrm{K}-\infty,|w|+i\mathrm{J}} = x_{\mathrm{K}-\infty,|w|+i\mathrm{J}}$) and any step $t \in \mathrm{T}$, we have $F^t(y)_{\mathrm{J}0,\infty\mathrm{J}} = F^t(x)_{\mathrm{J}0,\infty\mathrm{J}}$ (resp. $F^t(y)_{\mathrm{K}-\infty,n\mathrm{J}} = F^t(x)_{\mathrm{K}-\infty,n\mathrm{J}}$). In other words, the information cannot be transmitted through cells that have known a blocking word as wide as the radius.

Remark 7. *If $(\Sigma \subset A^{\mathrm{M}}, F)$ is a cellular automaton of anchor $m \in \mathrm{N}$ and anticipation $n \in \mathrm{N}$, $i, j \in \mathrm{M}$, $k \geq m$, $l \geq n$, u an (i, k)-blocking word, v a (j, l)-blocking word, and w such that the concatenation uwv is in the language $\mathcal{L}(\Sigma)$, then uwv is $(i, |u|+i+|w|-j+l)$-blocking.*

The previous remark strongly uses the one-dimensional structure of the space to concatenate blocking words, as do the results following from it.

The next result comes essentially from [2].

Proposition 12. *Let (Σ, F) be a cellular automaton of radius r. Then F is equicontinuous if and only if there exists some $l \in \mathrm{N}$ such that any word of $\mathcal{L}_l(\Sigma)$ is r-blocking.*

Proof. If for any l we can find some word of A^l which is not $(\max(m,n))$-blocking, then by the increasing property of blocking words and compactness, we can build a configuration which does not admit any $(\max(m,n))$-blocking central pattern, which thus is not equicontinuous by Remark 6. Conversely, if any word of $u \in \mathcal{L}_l(\Sigma)$ is (i_u, r)-blocking and (j_u, r)-blocking for some $i_u, j_u \in \mathrm{Z}$, then any $q \in \mathrm{N}$ and any word of $w \in \mathcal{L}_{2l+q+\max_{u\in\mathcal{L}_l(\Sigma)} j_u}(\Sigma)$ is, by Remark 7, $(l + i_{w_{\mathrm{J}0,l\mathrm{J}}} + \left|w_{\mathrm{J}l,|w|-l\mathrm{J}}\right| - j_{w_{\mathrm{J}|w|-l,|w|\mathrm{J}}} + r)$-blocking, and in particular, thanks to the increasing property, q-blocking. All widths of blockingness are obtained by all sufficiently large words, and Remark 6 gives that all the configurations are equicontinuous. □

A dynamical system (X, F) is **nonwandering** (resp. **transitive**) if for any nonempty open set $U \subset X$ (resp. and $V \subset X$), there is some point $x \in U$ and some step $t \in \mathrm{T} \setminus \{0\}$ such that $F^t(x) \in U$ (resp. $F^t(x) \in V$). In the case of bijective systems, it is known that the definitions with $\mathrm{T} = \mathrm{Z}$ or $\mathrm{T} = \mathrm{N}$ coincide. Moreover, thanks to the cylinder base, it is easy to see that a subshift Σ is nonwandering (resp. transitive) if and only if for any word $u \in \mathcal{L}(\Sigma)$ (resp. and $v \in \mathcal{L}(\Sigma)$), there exists a word w such that $uwu \in \mathcal{L}(\Sigma)$ (resp. $uwv \in \mathcal{L}(\Sigma)$). Of course it is the case for the full shift, and for many others.

We can actually apply infinitely the previous remark. Let $u \in A^*$ and $U_l = \bigcup_{j>l}[u]_j$ for $l \in \mathrm{N}$. If Σ is nonwandering, then by induction (and thanks to compactness) the intersection $\bigcap_{l\in\mathrm{N}} U_l$ is nonempty. If Σ is transitive, then each U_l is dense and (thanks to Baire's theorem) so is the intersection $\bigcap_{l\in\mathrm{N}} U_l$. If $\mathrm{M} = \mathrm{Z}$, the same can be done with $V_l = \bigcup_{j>l}[u]_{-j}$ for $l \in \mathrm{N}$, and $\bigcap_{l\in\mathrm{N}} U_l \cap V_l$. Applying this to the blocking words, we get the well-known dichotomy between sensitivity and density of equicontinuous points.

Theorem 2. *Let (Σ, F) be a nonsensitive cellular automaton of radius r. If Σ is nonwandering, then F admits some equicontinuous configuration. Moreover, if Σ is transitive, then F is quasiequicontinuous.*

The converse comes directly from the definition: a sensitive system cannot have any equicontinuous point. Theorem 2 gives us the following.

Corollary 13. *If F is a sensitive cellular automaton, then it cannot have any r-blocking words, hence it is 2^r-sensitive.*

Proof of Theorem 2. Suppose that $u \in A^*$ is an (i,k)-blocking word for F, with $k \geq r$, and $U = \bigcap_{l\in\mathrm{N}} \bigcup_{j>l}[u]_j$ if $\mathrm{M} = \mathrm{N}$, $U = \bigcap_{l\in\mathrm{N}} \left(\bigcup_{j>l}[u]_j \cap \bigcup_{j>l}[u]_{-j}\right)$ if $\mathrm{M} = \mathrm{Z}$. By Remarks 7 and 6, any configuration of U is equicontinuous. We have already seen that such a set U is nonempty if Σ is nonwandering, and is dense if Σ is transitive. □

Preperiodicity. We say that a dynamical system (X, F) is **preperiodic** if there exist a period $p \in \mathrm{N} \setminus \{0\}$ and a preperiod $q \in \mathrm{N}$ such that $F^{q+p} = F^q$. In particular, if, besides, F is surjective, then F is periodic. It is also well known that, for subshifts, this condition is equivalent to finiteness.

Remark 8. *The stability of a configuration x of some symbolic system (Λ, F) can be expressed in terms of traces: x is 2^{-k}-stable, with $k \in \mathrm{N}^*$, if and only if there exists some radius $l \in \mathrm{N}$ such that $T_F^{\langle k\rangle}([x_{\langle l\rangle}])$ is a singleton. Similarly, F is equicontinuous if and only* *if for any radius $k \in \mathbb{N}^*$ there exists another radius $l \in \mathbb{N}^*$ such that for any word $u \in A^{\langle l \rangle}$, the trace $T_F^{\langle k \rangle}([u])$ is a singleton.*

Proposition 14. *A symbolic system (Λ, F) is equicontinuous if and only if all of its traces are finite.*

Proof. Let F be an equicontinuous symbolic system and $k \in \mathbb{N}$. There exists a radius $l \in \mathbb{N}$ such that for any $u \in A^{\langle l \rangle}$, $T_F^{\langle k \rangle}([u])$ is a singleton. Consequently, $\left|\tau_F^{\langle k \rangle}\right| = \left|\bigcup_{u \in A^{\langle l \rangle}} T_F^{\langle k \rangle}([u])\right| \leq |A^{\langle l \rangle}|$.
Conversely, if $\tau_F^{\langle k \rangle}$ is finite, then, as a subshift, it is (p, q)-preperiodic, for some $p \in \mathbb{N} \setminus \{0\}$ and $q \in \mathbb{N}$. Any point $x \in \Lambda$ is ε-stable, since any point y of the neighborhood $\bigcap_{t<p+q} F^{-t}(\mathcal{B}_\varepsilon(F^t(x)))$ satisfies $\forall t \in \mathbb{N}, \mathrm{d}(F^t(x), F^t(y)) < \varepsilon$. □

In the case of cellular automata, all the traces have as a projection the trace of width 1. Consequently, the period and preperiod are uniform on all the traces, which slightly generalizes a classical result [2, 4].

Corollary 15. *Any cellular automaton is equicontinuous if and only if it is preperiodic.*

Proof. Let F be an equicontinuous cellular automaton on a subshift $\Sigma \subset A^{\mathbb{M}}$. From Proposition 14, τ_F^1 is preperiodic, *i.e.* there are $p \in \mathbb{N} \setminus \{0\}, q \in \mathbb{N}$ such that for any configuration $x \in \Sigma$, $F^{p+q}(x)_0 = F^q(x)_0$. We conclude by strong shift-invariance.
Conversely, it is well known that any preperiodic dynamical system is equicontinuous. □

Regarding sensitivity, it is transmitted from the system to all of its sufficiently fine traces.

Proposition 16. *Let (Σ, F) a ε-sensitive symbolic system and $\mathcal{P}$ a partition of diameter less than ε. Then $\tau_F^{\mathcal{P}}$ is a sensitive subshift.*

Proof. Let $x \in \Sigma$ and $\delta > 0$. By continuity, there exists $\delta' > 0$ such that for any configuration $y \in \mathcal{B}_{\delta'}(x)$, we have $\mathrm{d}(T_F^{\mathcal{P}}(x), T_F^{\mathcal{P}}(y)) < \delta$. The sensitivity of F gives us some configuration $y \in \mathcal{B}_{\delta'}(x)$ and some step $t \in \mathbb{T}$ such that $\mathrm{d}(F^t(x), F^t(y)) > \varepsilon$. Since $\mathcal{P}$ has a smaller diameter, we get $T_F^{\mathcal{P}}(x)_t \neq T_F^{\mathcal{P}}(y)_t$, *i.e.* $\mathrm{d}(\sigma^t T_F^{\mathcal{P}}(x), \sigma^t T_F^{\mathcal{P}}(y)) = 1$, with $\mathrm{d}(T_F^{\mathcal{P}}(x), T_F^{\mathcal{P}}(y)) < \delta$. □

5. Expansivity

Expansivity represents a very strong instability property: the tiniest difference between two initial points will eventually become big in their evolution. Let (X, F) be a dynamical system and $\varepsilon > 0$. F is ε-**expansive** if for any two points $x \neq y \in X$, there exists some step $t \in \mathbb{T}$ at which $\mathrm{d}(F^t(x), F^t(y)) > \varepsilon$.

It is not difficult to see that, for symbolic systems, this property implies that $T_F^{\mathcal{P}}$ is injective whenever $\mathcal{P}$ is a partition of diameter less than ε. Any of these partitions can be seen as a generator by itself. In other words, the system is conjugate to $\tau_F^{\mathcal{P}}$. Of course, subshifts are expansive, and expansivity is a topological notion, so we get that the expansive systems are essentially the subshifts.

We say that a symbolic system (Σ, F), with $\Sigma = A^{\mathbb{M}}$, is **right-expansive** with **width** $k \in \mathbb{N} \setminus \{0\}$ if for any two configurations $x, y \in \Sigma$ such that $x_{\llbracket 0,\infty\llbracket} \neq y_{\llbracket 0,\infty\llbracket}$, there exists a step $t \in \mathbb{T}$ with $F^t(x)_{\llbracket 0,k\llbracket} \neq F^t(y))_{\llbracket 0,k\llbracket}$. If $\mathbb{M} = \mathbb{N}$, then this is equivalent to expansivity. Otherwise, we can symmetrically define **left-expansivity**. We say that a cellular automaton is expansive with **width** k if it is right-expansive and left-expansive with width k; this coincides with the definitions over dynamical systems.

Note that if (Σ, F) is right-expansive with width k, then for any $l \geq k$, the trace τ_F^l is conjugate to τ_F^k via the projection $\pi_{\llbracket 0,k\llbracket}$. Otherwise there would exist x, y with $T_F^k(x) = T_F^k(y)$ but $T_F^l(x) \neq T_F^l(y)$ and thus $x \neq y$. In other particular, if $\mathbb{M} = \mathbb{Z}$, we can see that the family of clopen partitions $(\left\{ [u] \middle| u \in A^{\llbracket -i,k\llbracket} \right\})_{i\in\mathbb{N}}$ composed of the cylinders arbitrarily large to the left, but bounded on the right, represents a generator. Of course, the same is true in the left-expansive case.

Proposition 17. *Any right-expansive (resp. left-expansive) cellular automaton of anchor* $m \in \mathbb{N}$ *and anticipation* $n \in \mathbb{N}$ *on some surjective SFT of order* n *(resp.* m*) is right-expansive with width* n *(resp. left-expansive with width* m*).*

Proof. Let (Σ, F) be such a cellular automaton and assume that there are two configurations $x, y \in \Sigma$ and some cell $i \in \mathbb{N}$ such that $x_i \neq y_i$ but $T_F^n(x) = T_F^n(y)$. Let $k \in \mathbb{N}$. Since Σ is a surjective SFT of order n, there exist two configurations $x^k \in \sigma^{-k}(x)$ and $y^k \in \sigma^{-k}(y)$ such that for any cells $i < n$, $x_i^k = y_i^k$. These two configurations are distinct but their traces $T_F^{k+n}\sigma^{-k}(x) = T_F^{k+n}\sigma^{-k}(y)$ are equal thanks to Proposition 6; hence the cellular automaton is not right-expansive. The left side can be proved symmetrically. □

Combining the left and right sides, we can bound the width of expansivity for cellular automata F by its radius r; the trace map T_F^r whose width is the radius is then bijective and F is conjugate to τ_F^r.

Expansivity of cellular automata represents a large domain of open questions, which is, in particular, the base of the justification of Nasu's textile systems [5]. This theory allowed to prove (*cf* [6, 7]) that any expansive cellular automaton over sofic mixing subshifts is conjugate to an SFT (equivalently, τ_F^r is an SFT). In the case when $\mathbb{T} = \mathbb{N}$, it is even conjugate to a full shift (of the form $B^{\mathbb{N}}$, where B is an alphabet).

6. Entropy

The **entropy** of some subshift Σ can be defined in the following way:

$$\mathcal{H}(\Sigma) = \lim_{n\to\infty} \frac{\log |\mathcal{L}_n(\Sigma)|}{n} .$$

The **entropy** of some symbolic system (X, F) can be defined in the following way:

$$\mathcal{H}(F) = \lim_{k\to\infty} \mathcal{H}(\tau_F^{\langle k\rangle}) = \lim_{k\to\infty} \lim_{n\to\infty} \frac{\log \left| \mathcal{L}_n(\tau_F^{\langle k\rangle}) \right|}{n} .$$

If F is a cellular automaton with anchor m and anticipation n, then the overlapping gives us the expression:

$$\mathcal{H}(F) = \lim_{k\to\infty} \mathcal{H}((\tau_F^{m+n+1})^{[k]}) .$$

If $k \in \mathrm{N}$, then the trace $\tau_F^{\langle 2k\rangle}$ of width $2k$ is essentially included in $(\tau_F^{\langle k\rangle})^2$; we get the very rough bound: $\mathcal{H}(\tau_F^{\langle 2k\rangle}) \leq 2\mathcal{H}(\tau_F^{\langle k\rangle})$. On the other hand, it is well known that entropy does not increase through factor maps: $\mathcal{H}(F) \geq \mathcal{H}(\tau_F^{\langle k\rangle})$, which gives the following fact.

Remark 9. *A cellular automaton has null entropy if and only if each of its traces does.*

The entropy of a cellular automaton can also be bounded by that of its right and left canonical factors.

Proposition 18. *Any cellular automaton* (Σ, F) *of anchor and anticipation* $m, n \in \mathrm{N}$ *has an entropy which is less or equal to the sum of that of its left and right canonical factors:*

$$\mathcal{H}(F) \leq \mathcal{H}(\tau_F^m) + \mathcal{H}(\tau_F^n)\ .$$

Proof. Let $l \in \mathrm{N}$ and $k \geq m+n$. Then by Proposition 7, there is an injection:

$$\begin{array}{rrcl} \phi : & \mathcal{L}_l(\tau_F^k) & \to & \mathcal{L}_l(\tau_F^m) \times \mathcal{L}_{k-m-n}(\Sigma) \times \mathcal{L}_l(\tau_F^n) \\ & z = (z^j)_{0\leq j<l} & \mapsto & (\pi_{\mathrm{J}0,m\mathrm{J}}(z), z^0_{\mathrm{J}m,k-n\mathrm{J}}, \pi_{\mathrm{J}k-n,k\mathrm{J}}(z))\ , \end{array}$$

which gives the following cardinality inequality:

$$\left|\mathcal{L}_l(\tau_F^k)\right| \leq \left|\mathcal{L}_l(\tau_F^m)\right| \left|\mathcal{L}_{k-m-n}(\Sigma)\right| \left|\mathcal{L}_l(\tau_F^n)\right|\ .$$

Taking the logarithm, and deleting the term which is negligible with respect to l, we get the entropy relatively to any partition into cylinders of width $k \geq m+n$:

$$\mathcal{H}(\tau_F^{\mathrm{J}0,k\mathrm{J}}) \leq \mathcal{H}(\tau_F^m) + \mathcal{H}(\tau_F^n)\ . \quad \square$$

In particular, for a cellular automaton of radius r, we get the bounds: $\mathcal{H}(\tau_F^r) \leq \mathcal{H}(F) \leq 2\mathcal{H}(\tau_F^r)$.

In the onesided case, we find again the following known equality.

Corollary 19 ([8]). *A CA* (Σ, F) *of anchor* 0 *and anticipation* $r \in \mathrm{N}$ *has an entropy which is equal to that of its canonical factor:*

$$\mathcal{H}(F) = \mathcal{H}(\tau_F^r)\ .$$

The same is true for expansive cellular automata, since they are conjugate to their canonical factor. Moreover, we saw in the previous section that any trace $\tau_F^{\mathrm{J}0,k\mathrm{J}}$ of width k greater than the radius r is conjugate to the canonical factor τ_F^r, which gives us that any cellular automata which are expansive in some side has the same entropy than its canonical factor

Proposition 20. *A left-expansive (resp. right-expansive) cellular automaton* F *of radius* $r \in \mathrm{N}$ *has entropy* $\mathcal{H}(F) = \mathcal{H}(\tau_F^r)$.

In particular, we find the result of [9]: any permutive cellular automaton (whose local rule acts as a permutation over the first or last cell of the neighborhood) is either onesided or expansive in some side, hence its entropy is equal to that of its canonical factor.

This simplification of the entropy expression in particular cases brings a natural hope of generalization.

Question 1 ([9]). *Does there exist a (computable) width* $k \in \mathrm{N}$ *such that* $\mathcal{H}(F) = \mathcal{H}(\tau_F^{\langle k\rangle})$*?*

Conclusion

Cellular automata are precisely the dynamical systems which can be defined from one of their factor subshifts: each cellular automaton is the inverse limit of overlaps of some subshift. It shares many properties with it, from the extreme case of expansivity, when it exactly behaves like the subshift, to equicontinuity or preperiodicity, when the subshift satisfies so strong properties that it constrains the whole system.

It is now natural to ask which other properties of the global system can be observed in the dynamics of the subshift itself. For instance, this article did not deal much with transitivity and its variants (mixingness...), which are preserved by factor maps. Hence any trace of a transitive cellular automaton is transitive. But, conversely, if F is a cellular automaton of anchor m and anticipation n, need F be transitive whenever its canonical factor is? Other such questions can be asked: is the entropy of F equal to that of this trace? does it factor onto some other cellular automaton G whenever the two corresponding traces do?

Cellular automata can also be defined over higher-dimensional networks, such as $\mathbb{Z}^2$, or the Cayley graph of any monoid. Everything in the article can be generalized to virtually cyclic monoids or groups, but in more complex contexts, the canonical factor does not play the same role, since it does not disconnect the space of cells. Indeed, the ability to stick together two (or a finite number of) parts of space-time diagrams to get a new one is the crucial point which makes one-dimensional dynamics so particular. It is a base argument in many other results [10, 11], whose generalization remains open in higher-dimensional cases. It is already known that the density and existence of equicontinuous points (Theorem 2) knows a two-dimensional counter-example [12].

References

[1] Gustav Arnold Hedlund. Endomorphisms and automorphisms of the shift dynamical system. *Mathematical Systems Theory*, **3**:320–375, 1969.

[2] Petr Krka. Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory & Dynamical Systems*, **17**:417–433, 1997.

[3] Franois Blanchard and Alejandro Maass. Dynamical properties of expansive one-sided cellular automata. *Israel Journal of Mathematics*, **99**:149–174, 1997.

[4] Alberto Dennunzio, Pierre Guillon, and Benot Masson. Stable dynamics of sand automata. In Giorgio Ausiello, Juhani Karhumki, Giancarlo Mauri, and Luke Ong, editors, 5th *IFIP International Conference on Theoretical Computer Science (TCS'08)*, volume 273 of *International Federation for Information Processing* , pages 157–169, Milan, Italie, September 2008. Springer, Boston.

[5] Masakazu Nasu. *Textile Systems for Endomorphisms and Automorphisms of the Shift* , volume 114 of *Memoirs of the American Mathematical Society* . American Mathematical Society, Providence, Rhode Island, March 1995.

[6] Petr Krka. *Topological and symbolic dynamics*. Socit Mathmatique de France, 2003.

[7] Masakazu Nasu. The dynamics of expansive invertible onesided cellular automata. *Transactions of the American Mathematical Society* , **354**(10):4067–4084, 2002.

[8] Franois Blanchard. Cellular automata and transducers: a topological view. In *Cellular Automata, Dynamical Systems and Neural Networks* . Kluwer, Dordrecht, 1994.

[9] Pietro di Lena and Luciano Margara. Row subshifts and topological entropy of cellular automata. *Journal of Cellular Automata* , **2**:131–140, 2007.

[10] Pierre Guillon and Gatan Richard. Nilpotency and limit sets of cellular automata. In Edward Ochmaski and Jerzy Tyszkiewicz, editors, 33^{rd} *International Symposium on the Mathematical Foundations of Computer Science (MFCS'08)* , volume 5162 of *Lecture Notes in Computer Science*, pages 375–386, Toru, Pologne, August 2008. Springer-Verlag.

[11] Pietro di Lena. Decidable properties for regular cellular automata. In Gonzalo Navarro, Leopoldo E. Bertossi, and Yoshiharu Kohayakawa, editors, 4^{th} *IFIP International Conference on Theoretical Computer Science (TCS'06)* , volume 209 of *International Federation for Information Processing* , pages 185–196, Santiago, Chili, August 2006. Springer-Verlag.

[12] Mathieu Sablik and Guillaume Theyssier. Topological dynamics of 2D cellular automata. In Arnold Beckmann, Costas Dimitracopoulos, and Benedikt Lwe, editors, *Logic and Theory of Algorithms*, 4^{th} *Conference on Computability in Europe (CiE'08)*, volume 5028 of *Lecture Notes in Computer Science*, Athnes, Grce, June 2008. Springer-Verlag.

INDEX