Bruno Durand (Ed)

JAC 2008

Journées Automates Cellulaires
Uzès, France, April 21 – April 25, 2008
Proceedings

Москва
Издательство МЦНМО
2008

Volume Editor

Bruno Durand
LIF, Aix-Marseille Université, CNRS
39 rue Joliot-Curie, 13013 Marseille, France
E-mail: Bruno.Durand@lif.univ-mrs.fr

Брюно Дюран (ред.) JAC 2008. Труды конференции по клеточным автоматам
(Uzés, Франция, 21–25 апреля 2008 г.)

ISBN 978-5-94057-377-7 MCCME Publishing House Moscow
Подписано в печать с диапозитивов 10.04.2008
Технический редактор Н. Олланже
Бумага белая. Печать офсетная. Гарнитура компьютер модерн
Издательство Московского центра непрерывного математического образования
119002, Москва, Большой Власьевский пер., 11
Отпечатано в типографии Эколь Нормаль Супериор, Лион, Франция

Printed at École Normale Supérieure Lettres et Sciences Humaines, Lyon, France
Typesetting: Camera-ready by author, volume composing by N. Ollinger, LIF, Marseille, France

Foreword

This volume contains papers and abstracts presented at the First Symposium on Cellular Automata "Journées Automates Cellulaires" (JAC 2008). This year's conference was held on 21-25 April 2008 at the Hôtel du Général d'Entraigues, in the scenic old city of Uzès, France.

The Symposium on Cellular Automata "Journées Automates Cellulaires" (JAC 2008) is an international conference centered on cellular automata and other models of computation. Topic of interest included: cellular automata, tilings, computation models, logic and other bistroodles.

The call for papers led to approximately 25 submissions from 7 countries. Each was assigned to at least two program committee members, refereed directly by them or treated by external referees. The program committee selected 14 regular and 5 exploratory papers. As chair of this committee, I would like to sincerely thank its members and the external referees for the valuable work they have invested in the reviewing process. In the name of both the program committee and the organizing committee, I would like to thank authors of selected papers: they chose JAC, a new conference, to submit very good papers that could have been also accepted at top notch and well established conferences with a wide scope.

During the four days of the conference 14 regular papers and 5 exploratory papers were presented. Indeed, the PC committee of JAC 2008 has created a special category for exploratory papers: we considered in this category papers dealing with very special new variants of cellular automata, papers that are scope borderline, or papers that present interesting works for the conference audience but are yet in preliminary state.

In addition to these presentations, 5 informal talks were also given. The extended abstracts of the invited talks, the regular and exploratory papers and the abstracts of the informal talks are included in these proceedings.

It was a pleasure and an honor to have the opportunity to organize this first Symposium on Cellular Automata "Journées Automates Cellulaires" (JAC 2008). The conference benefited from six invited speakers, whose topics spanned the above list, and a total of 50 registered attendees. I would like to express my thanks to the six invited speakers, Jacques Mazoyer, Gianpiero Cattaneo, Eric Goles, Jarkko Kari, Maurice Nivat and Alexander Shen.

I wish to particularly thank the sponsors for the meeting: Laboratoire d'Informatique Fondamentale de Marseille, Escape research group of the Laboratoire d'Informatique Fondamentale de Marseille, Laboratoire de l'Informatique du Parallélisme, IXXI Rhône-Alpes Complex Systems Institute and Université de Provence.

Special thanks are due to A. Voronkov for his EasyChair software (`http://www.easychair.org/`) which gives the organizers of conferences a remarkable level of comfort. The conference could not be organized without the help of the members of the Organizing Committee, chaired by Grégory Lafitte and Nicolas Ollinger. I wish to express in these lines my deepest gratitude for the harsh work done mainly in a vacation period and for their tenacity.

These proceedings are published in an electronic format and in a printed version with ISBN 978-5-94057-377-7. The electronic proceedings are available through several portals, and in particular through HAL. HAL is an electronic repository managed by several French research agencies. We want to thank HAL for hosting the proceedings of JAC and guaranteeing them perennial availability. The rights on the articles in the proceedings are kept with the authors and the papers are available freely, under a Creative Commons license.

Marseille, April 2008 Bruno Durand

President of the Conference

Jacques Mazoyer

Program Committee

Jean-Paul Allouche
Marianne Delorme
Bruno Durand
Alejandro Maass
Nikolai Vereshchagin

Organizing Committee

Grégory Lafitte *(co-chair)*
Nicolas Ollinger *(co-chair)*
Éric Remila *(pôle lyonnais)*
Alexis Ballier
Emmanuel Jeandel
Benoît Masson
Victor Poupet
Gaétan Richard

External Referees

Julio Aracena
Alexis Ballier
Julien Cervelle
Marianne Delorme
Jérôme Durand-Lose
Olivier Finkel
Emmanuel Jeandel
Jacques Mazoyer
Andrés Moreira
Francis Oger
Nicolas Ollinger
Sylvain Perifel
Victor Poupet
Éric Remila
Gaétan Richard
Zsuzsanna Róka
Andrei Romashchenko
Andrey Rumyantsev
Alexander Shen
Véronique Terrier
Jean-Baptiste Yunès

Table of Contents

Other Talks: Abstracts and Extended Abstracts

A FULL CELLULAR AUTOMATON TO SIMULATE PREDATOR-PREY SYSTEMS COMPARISON WITH THE DISCRETE TIME LOTKA-VOLTERRA EQUATIONS

G. CATTANEO

Università degli Studi di Milano-Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione,
Via Bicocca degli Arcimboldi 8, 20126 Milano (Italy)
E-mail address: cattang@disco.unimib.it

Abstract. A Cellular Automata (CA) model describing a predator-prey dynamics is introduced; this model is based on a uniformly applied update rule which is fully local, i.e., without any spurious Monte Carlo step during the diffusion phase. A particular attention has been addressed to the comparison of the obtained discrete time simulations with the theoretical results from the difference equation versions of some generalized version of the Lotka-Volterra equation.

COMMUNICATION COMPLEXITY AND CELLULAR AUTOMATA

E. GOLES

E-mail address: eric.chacc@uai.cl

Abstract. Given a one dimensional CA with two states, nearest interactions and n left bits, x, a central bit c and n right bits, y. The question is what is the minimum information the left (say Alice) have to send to the right (say Bob) in order to Bob compute the n-th step state of the C.A? We will give examples, numerical experiments and some exact results.

Journées Automates Cellulaires 2008 (Uzès), pp. 3-14

UNDECIDABLE PROPERTIES ON THE DYNAMICS OF REVERSIBLE ONE-DIMENSIONAL CELLULAR AUTOMATA

JARKKO KARI

Department of Mathematics
FIN-20014 University of Turku
Finland
E-mail address: jkari@utu.fi

Abstract. Many properties of the dynamics of one-dimensional cellular automata are known to be undecidable. However, the undecidability proofs often rely on the undecidability of the nilpotency problem, and hence cannot be applied in the case the automaton is reversible. In this talk we review some recent approaches to prove dynamical properties of reversible 1D CA undecidable. Properties considered include equicontinuity (=periodicity), sensitivity, variants of mortality, one-sided expansivity and regularity. All these properties are undecidable, according to recent proofs obtained in collaboration with N.Ollinger or V.Lukkarila.

Introduction

Let S be a finite set – the state alphabet – and consider bi-infinite sequences of symbols of S, called configurations over S. The set $S^{\mathbb{Z}}$ of all configurations is endowed with the standard compact and metrizable topology obtained as the countably infinite product of the discrete topology on S. A one-dimensional cellular automaton (CA for short) over state set S is a continuous transformation $G : S^{\mathbb{Z}} \longrightarrow S^{\mathbb{Z}}$ that commutes with the shift function $\sigma : S^{\mathbb{Z}} \longrightarrow S^{\mathbb{Z}}$, defined by $\sigma(x)_i = x_{i+1}$ for all $x \in S^{\mathbb{Z}}$ and $i \in \mathbb{Z}$.

It follows from the classical Garden-of-Eden theorem that every injective CA transformation G is also surjective, and hence bijective. Because $S^{\mathbb{Z}}$ is compact, the inverse function G^{-1} is continuous and commutes with the shift — it is also a CA. We call an injective G a reversible cellular automaton, and G^{-1} its inverse automaton. See [6] for more details and further historical results.

To treat cellular automata algorithmically one needs a finite representation for them. This is provided by the famous Curtis-Hedlund-Lyndon theorem, which states that cellular automata are exactly the functions that can be defined by a simultaneous, parallel application of a local update rule [3]. More precisely, a non-negative integer r (called the

Key words and phrases: cellular automata.
Research supported by the Academy of Finland grant 211967.

neighborhood radius) and a local rule $f : S^{2r+1} \longrightarrow S$ define the CA function G over state set S where

$$G(x)_i = f(x_{i-r}, \ldots, x_{i+r})$$

for all $i \in \mathbb{Z}$. All cellular automata functions G arise in this fashion.

Many dynamical properties of cellular automata are known to be undecidable: there is no algorithm that would determine if a given CA (given in terms of its local update rule) has the property. Examples of such undecidable properties include nilpotency, equicontinuity and sensitivity (see Section 3 below). These properties are known to be undecidable among the general, not necessarily reversible cellular automata. The undecidability proofs rely mostly on nilpotency, which is a property possessed only by non-reversible CA. Until recently, no analogous undecidability results were known for reversible CA. Yet reversible CA constitute an important family of dynamical systems whose properties and long time behavior should be better understood. In this paper we report some recent undecidability results for a few dynamical properties for reversible CA. The results are obtained with N.Ollinger and V.Lukkarila. We give the results without proofs here — the proofs will appear elsewhere.

The paper is organized as follows: in Section 1 we define and discuss the dynamical properties studied in this paper. In Section 2 we briefly recall some concepts and results on Wang tiles. Section 3 reviews some basic undecidability results for non-reversible CA, and in Section 4 we give undecidability results for reversible CA.

1. Dynamical properties

This paper summarizes some undecidability results on dynamical properties of reversible one-dimensional cellular automata. Properties considered are equicontinuity (=periodicity), sensitivity, variants of mortality, one-sided expansivity and regularity. In this section we define these concepts.

First some general terms and notations: elements $i \in \mathbb{Z}$ are referred to as cells. For configuration $x \in S^{\mathbb{Z}}$ and integers $m \leq n$ we use the notation $x_{[[m,n]]}$ to denote the word $x_m x_{m+1} \ldots x_n$. Configuration x is called spatially periodic if $\sigma^m(x) = x$ for some $m > 0$. Spatially periodic configurations form a dense subset of $S^{\mathbb{Z}}$. Configuration x is called temporally periodic (or just periodic) for G if $G^p(x) = x$ for some period $p > 0$, and it is called (temporally) eventually periodic for G if $G^{m+p}(x) = G^m(x)$ for some pre-period $m \geq 0$ and period $p > 0$. A spatially periodic configuration is always eventually periodic. In reversible CA eventually periodic configurations are all periodic, so in reversible CA periodic configurations are dense. A well known open problem asks whether the same holds for all surjective CA.

1.1. Equicontinuity points

Configuration x is an *equicontinuity point* for G if for all $m \in \mathbb{N}$ there exists $M \in \mathbb{N}$ such that for all configurations y

$$x_{[[-M,M]]} = y_{[[-M,M]]} \Longrightarrow G^i(x)_{[[-m,m]]} = G^i(y)_{[[-m,m]]} \text{ for all } i \in \mathbb{N}.$$

The evolution of equicontinuity points can be reliable simulated up to any precision on a computer: all forthcoming states within an observation window of radius m are uniquely determined by the initial states within a window of radius M.

If G is reversible one may be interested in the evolution of x both in backwards and forwards in time. Let us define *two-way equicontinuity* of x analogously, only difference being that time i now takes also negative values: for all m there exists M such that for all $y \in S^{\mathbb{Z}}$

$$x_{[[-M,M]]} = y_{[[-M,M]]} \Longrightarrow G^i(x)_{[[-m,m]]} = G^i(y)_{[[-m,m]]} \text{ for all } i \in \mathbb{Z}.$$

The two definitions, however, coincide:

Lemma 1.1. *Let G be reversible. Point $x \in S^{\mathbb{Z}}$ is an equicontinuity point for G if and only if it is a two-way equicontinuity point for G.*

Proof. Let x be a configuration that is not a two-way equicontinuity point for reversible G. Then for some window radius m holds the following: for all M there is some configuration y and time $i \in \mathbb{Z}$ such that $x_{[[-M,M]]} = y_{[[-M,M]]}$ while $G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]}$. Periodic configurations are dense, so there are periodic x', y' such that

$$x'_{[[-M,M]]} = x_{[[-M,M]]} = y_{[[-M,M]]} = y'_{[[-M,M]]}$$

and

$$G^i(x')_{[[-m,m]]} = G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]} = G^i(y')_{[[-m,m]]}.$$

Since x' and y' are temporally periodic, we have a positive j such that

$$G^j(x')_{[[-m,m]]} = G^i(x')_{[[-m,m]]} \neq G^i(y')_{[[-m,m]]} = G^j(y')_{[[-m,m]]}.$$

Then, either $G^j(x')_{[[-m,m]]} \neq G^j(x)_{[[-m,m]]}$ or $G^j(y')_{[[-m,m]]} \neq G^j(x)_{[[-m,m]]}$. In any case, a configuration contradicting the equicontinuity of x exists for all M, that is, x is not equicontinuous.

The other direction is trivial. ■

1.2. Equicontinuity

Cellular automaton G is called *equicontinuous* if all configurations are equicontinuity points. It easily follows from the compactness of $S^{\mathbb{Z}}$ that the radius M of the initial window can be chosen independently of configuration x, that is, G is equicontinuous iff the following holds:

$$(\forall m \in \mathbb{N})(\exists M \in \mathbb{N})(\forall x, y \in S^{\mathbb{Z}})$$
$$x_{[[-M,M]]} = y_{[[-M,M]]} \Longrightarrow G^i(x)_{[[-m,m]]} = G^i(y)_{[[-m,m]]} \text{ for all } i \in \mathbb{N}.$$

For reversible G one can define *two-way equicontinuity* by considering also negative time $i < 0$ (that is, by requiring all configurations to be two-way equicontinuity points) but according to Lemma 1.1 the concept obtained would be equivalent to normal equicontinuity.

A CA is called *periodic* (*eventually periodic*) if all configurations are periodic (eventually periodic, respectively). One can easily see that the pre-period m and period p can be then chosen independently of the configuration x, so that a CA is periodic (eventually periodic) if and only if there exists p (or m and p) such that $G^p(x) = x$ (or $G^{m+p}(x) = G^m(x)$, respectively) for all x.

It is known that one-dimensional equicontinuous cellular automata are exactly the eventually periodic CA, and among surjective CA equicontinuity is equivalent to periodicity [12, 2]. In particular, all surjective, equicontinuous CA are reversible. These facts are summarized in the following theorem:

Theorem 1.2. *The following hold* [12, 2]*:*

(a) *One-dimensional CA is equicontinuous if and only if it is eventually periodic.*
(b) *One-dimensional surjective CA is equicontinuous if and only if it is periodic.*

A cellular automaton is called *nilpotent* if there exists a positive integer i such that $G^i(S^{\mathbb{Z}})$ contains only one element. Nilpotent CA have the most trivial dynamics possible: all activity dies out within time i. It is clear that a nilpotent CA is eventually periodic, and hence equicontinuous.

1.3. Sensitivity

A CA is *sensitive*, if there is an observation window radius $m \in \mathbb{N}$ such that for every initial configuration x and every $M \in \mathbb{N}$ there exists a configuration y and time $i \in \mathbb{N}$ such that $x_{[[-M,M]]} = y_{[[-M,M]]}$ but $G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]}$. In other words, every configuration x can be modified arbitrarily far away in space in such a way that the modification eventually propagates inside the observation window. Note that the definition states that in sensitive CA no configuration is equicontinuous, and moreover, the observation window size m that contradicts the equicontinuity at point x can be chosen independently of x. It turns out that the second condition is automatically satisfied: If a one-dimensional CA has no equicontinuity points then arbitrarily distant modifications can be made to any configuration in such a way that the change propagates into the observation window whose radius m equals the neighborhood radius r of the CA [12]. So we have the following:

Theorem 1.3. *A one-dimensional CA is sensitive if and only if it has no equicontinuity points* [12].

A CA is called *almost equicontinuous* if it has some equicontinuity points. Almost equicontinuity and sensitivity are complementary properties.

Two-way sensitivity of a reversible CA is defined analogously to sensitivity, only difference being that the time i when $G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]}$ may be negative. It is easy to see, analogously to Theorem 1.3, that a reversible one-dimensional CA is two-way sensitive if and only if it has no two-way equicontinuity points. It follows then from Lemma 1.1 that sensitivity and two-way sensitivity are equivalent concepts among reversible CA.

1.4. Expansivity

CA G is *positively expansive* if any difference in configurations eventually propagates inside a fixed observation window: there exists $m \in \mathbb{N}$ such that

$$x \neq y \Longrightarrow G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]} \text{ for some } i \in \mathbb{N}.$$

A reversible CA is *expansive* if time i is allowed to have a negative value: there exists $m \in \mathbb{N}$ such that

$$x \neq y \Longrightarrow G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]} \text{ for some } i \in \mathbb{Z}.$$

According to our naming convention above, expansivity should be termed two-way expansivity, but we rather decided to follow the historical terminology. Unlike for the dynamical properties discussed so far, the concepts of expansivity and positive expansivity are not equivalent. For example, the shift function σ is expansive but not positively expansive. In fact, it is easy to see that a reversible CA can never be positively expansive.

Expansivity and positive expansivity of one-dimensional CA have quite natural left- and right-sided variants. Let us call a CA *positively left-expansive* if there exists $m \in \mathbb{N}$ such that

$$x_{[[0,\infty)} \neq y_{[[0,\infty)} \Longrightarrow G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]} \text{ for some } i \in \mathbb{N}.$$

Analogously, in *positively right-expansive* CA differences propagate to the right. A CA is clearly positively expansive iff it is both positively left- and positively right-expansive. The shift σ is an example of a positively left expansive CA. For reversible CA we analogously define *left-* and *right-expansive* CA by allowing time i to obtain also negative values: G is left-expansive if for some $m \in \mathbb{N}$ holds

$$x_{[[0,\infty)} \neq y_{[[0,\infty)} \Longrightarrow G^i(x)_{[[-m,m]]} \neq G^i(y)_{[[-m,m]]} \text{ for some } i \in \mathbb{Z}.$$

We see in Theorem 4.4 below that it is undecidable whether a given reversible CA is left-expansive.

1.5. Equicontinuity classification of CA

In [12] Kurka proposed a classification of one-dimensional cellular automata into four classes based on their degree of sensitivity to initial conditions. The classes are as follows:

(K1) Equicontinuous CA.
(K2) Almost equicontinuous CA that are not equicontinuous.
(K3) Sensitive CA that are not positively expansive.
(K4) Positively expansive CA.

Each 1D CA belongs to exactly one of these classes. First three classes are based on the number of equicontinuity points (none, some but not all, and all configurations are equicontinuity points, respectively). No reversible CA is in class (K4), so when classifying reversible CA it makes sense to replace (K3) and (K4) by

(K3') Sensitive CA that are not expansive.
(K4') Expansive CA.

1.6. Mortality

Mortality of a dynamical system refers to the property that the evolution leads from every initial configuration into an "accepting" configuration. In cellular automata theory, acceptance can be defined in various ways which leads to different variants of mortality. We consider two variants.

Let us specify a set $F \subseteq S$ of accepting states. In our first variant acceptance happens when some cell enters an accepting state, and in the second variant acceptance happens when a fixed, predefined cell enters an accepting state. This leads to the following definitions: G is *globally mortal* with respect to set $F \subseteq S$ if for every $x \in S^{\mathbb{Z}}$ there are $i \in \mathbb{N}$ and $j \in \mathbb{Z}$ such that $G^i(x)_j \in F$. It is *locally mortal* with respect to set F if for every $x \in S^{\mathbb{Z}}$ there is $i \in \mathbb{N}$ such that $G^i(x)_0 \in F$. Every locally mortal CA is also globally mortal, while the converse is not true.

1.7. Column subshifts and the language classification of CA

In [12] also another classification of CA was proposed based on the complexity of their column subshifts. A column subshift is the set of all possible temporal sequences of views to the CA configurations through a fixed finite window. More precisely, let G be a one-dimensional CA and let k be a positive integer, the width of the observation window. The column subshift $\Sigma_k(G)$ of width k associated to G is

$$\Sigma_k(G) = \{y \in (S^k)^{\mathbb{N}} \mid \exists x \in S^{\mathbb{Z}} \ : \ y_i = G^i(x)_{[[1,k]]} \text{ for all } i \in \mathbb{N}\}.$$

It is a one-sided subshift over the alphabet S^k. Term *trace subshift* was used in [4] for the column subshift $\Sigma_1(G)$ of width one.

The column language $L_k(G)$ of width k is defined as the set of finite subwords of sequences in $\Sigma_k(G)$. It is a language over the alphabet S^k. Language $L_k(G)$ is always context-sensitive [14].

As in several sections above, if G is reversible then it makes sense to allow also negative time. The two-way column subshift of width k associated to G is

$$\{y \in (S^k)^{\mathbb{Z}} \mid \exists x \in S^{\mathbb{Z}} \ : \ y_i = G^i(x)_{[[1,k]]} \text{ for all } i \in \mathbb{Z}\}.$$

Among reversible (and even surjective) CA this does not, however, bring any new information on the dynamics because the language of this two-sided subshift is the same $L_k(G)$ extracted from the one-sided case.

Cellular automaton G is called *regular* if $\Sigma_k(G)$ is a sofic shift for every $k > 0$. This is equivalent to $L_k(G)$ being a regular language. Many questions on the dynamics of a given CA become decidable when restricted to regular CA [10]. This fact is largely based on the following result:

Theorem 1.4. *If G is regular then the column subshift $\Sigma_k(G)$ of width k can be effectively constructed, for every $k > 0$ [10].*

CA G is equicontinuous if and only if it is eventually periodic with some pre-period m and period p (see Theorem 1.2). This is equivalent to the column subshift $\Sigma_k(G)$ being bounded periodic: For all $y \in \Sigma_k(G)$ we have $y_i = y_{i+p}$ whenever $i \geq m$. It is easy to see that such subshifts are always sofic (even of finite type), so all equicontinuous CA are regular.

The language classification of CA proposed in [12] has three classes:

(L1) Equicontinuous CA (the column subshifts are all bounded periodic).
(L2) Regular but not equicontinuous CA (the column subshifts are sofic but not bounded periodic).
(L3) Non-regular CA (a column subshift is not sofic).

2. Wang tiles

Wang tiles and the tiling problem have been used in many undecidability proofs concerning cellular automata. While the tiling problem relates naturally to two-dimensional CA, also limiting behavior of one-dimensional CA can be treated by interpreting the space-time diagrams as plane tilings. This leads naturally to the definition of determinism in Wang tiles.

A Wang tile is an oriented unit square tile with labeled edges. We denote by $N(t)$, $E(t)$, $S(t)$ and $W(t)$ the label of the north, east, south and west edge of tile t, respectively.

A tile set T is a finite set of such tiles. A valid tiling by T is an assignment $c \in T^{\mathbb{Z}^2}$ of tiles on two-dimensional lattice in such a way that abutting edges of adjacent tiles are everywhere identical, that is, for every $(i,j) \in \mathbb{Z}^2$ we have $N(c(i,j)) = S(c(i,j+1))$ and $E(c(i,j)) = W(c(i+1,j))$.

The tiling problem is the algorithmic question of determining whether a given set T admits at least one valid tiling. This problem was proved undecidable by R.Berger in [13]. This is the starting point in many reductions to prove undecidability results for cellular automata.

To deal with one-dimensional CA we add the following constraint on tile sets: tile set T is *NW-deterministic* if the colors of the north and the west edges determine each tile uniquely, that is, for all $t, u \in T$

$$N(t) = N(u) \wedge W(t) = W(u) \Longrightarrow t = u.$$

We define analogously NE- SW- and SE-determinism. Tile set T is called *two-way deterministic* if it is both NW- and SE-deterministic (i.e. deterministic in two opposite directions), and T is *four-way deterministic* if determinism holds in all four directions.

If c is a valid tiling admitted by a NW-deterministic tile set T then any southwest-to-northeast diagonal of c uniquely determines the next diagonal below. A local rule using just two tiles of the previous diagonal gives each tile. Valid tilings become space-time diagrams of a one-dimensional CA if we interpret the diagonals as configurations of the CA. With a similar interpretation a two-way deterministic tile set yields a reversible CA. This method is better explained below in the proofs of Theorems 3.1 and 4.1.

The following result was proved in [15]. A weaker version dealing with NW-deterministic tile sets only was proved earlier in [5]. The result provides a basis for some of the undecidability results reported here:

Theorem 2.1. *It is undecidable whether a given 4-way deterministic tile set admits a valid tiling* [15].

3. Non-reversible CA

In this section we review some undecidability results concerning general, not necessarily reversible CA. Using the undecidability of the tiling problem among NW-deterministic tile sets (Theorem 2.1) we easily obtain the following:

Theorem 3.1. *It is undecidable whether a given one-dimensional CA is nilpotent* [5]. *The question is undecidable even among CA over the binary state set $S = \{0,1\}$* [1].

Proof. For a given NW-deterministic set T of Wang tiles we construct a one-dimensional CA over state set $S = T \cup \{q\}$ where $q \notin T$ is a new state. The local update rule only uses the state s_1 of the cell and state s_2 of its right neighbor. If $s_1, s_2 \in T$ and there exists a tile $t \in T$ such that

$$W(t) = E(s_1) \text{ and } N(t) = S(s_2)$$

then the new state of the cell is t. (Note that this t is unique due to NW-determinism.) In all other cases the new state is q. It is clear that the configuration ${}^{\omega}q^{\omega} = \ldots qqqq \ldots$ is a fixed point of this CA. Any southwest-to-northeast diagonal of a valid tiling is a configuration that never evolves to this fixed point, so if T admits a valid tiling then the CA is not nilpotent.

Conversely, if a valid tiling does not exist then, by a compactness argument, the $n \times n$ square can not be properly tiled for some n. This means that inside every segment of length n state q will appear within the first $2n$ time steps. As state q spreads the configuration becomes ${}^{\omega}q^{\omega}$ in at most $3n$ time steps. Hence the CA is nilpotent. The first result now follows from the undecidability of the tiling problem among NW-deterministic tile sets.

See [1] for the technique to reduce the state set into the binary alphabet. ■

Corollary 3.2. *It is undecidable whether a given 1D CA is equicontinuous* [1]. *It is undecidable whether it is sensitive* [1]. *These two properties are recursively inseparable. This is true even among CA over the binary state set $S = \{0, 1\}$.*

Proof. Let G be an arbitrary 1D CA of radius r. Then clearly $G \circ \sigma^{r+1}$ is nilpotent (and hence equicontinuous) if G is nilpotent, and it is sensitive if G is not nilpotent. ■

Among equicontinuity classification, the decidability status of positive expansivity remains a challenging open problem. Concerning the mortality of non-reversible CA, we have the following easy corollary:

Corollary 3.3. *Local mortality and global mortality are undecidable among one-dimensional CA.*

Proof. Let $F = \{q\}$ where q is the external non-tile state in the proof of Theorem 3.1. The CA constructed in the proof is locally (and globally) mortal with respect to F if and only if the tile set does not admit a valid tiling. ■

Finally, it was shown in [10] that it is undecidable to determine if a given one-dimensional CA is regular. In fact, nilpotency and non-regularity are recursively inseparable properties. In particular, this means that the classes (L1) and (L3) in the language classification of Section 1.7 are recursively inseparable from each other:

Corollary 3.4. *It is undecidable whether a given one-dimensional CA is regular* [10]. *More precisely, nilpotency and non-regularity are recursively inseparable.*

Proof. Suppose there exists an algorithm to separate nilpotent CA from non-regular CA. Then we can decide nilpotency of a given G as follows: The separating algorithm either tells that G is definitely not nilpotent or that G is definitely regular. In the first case we know the non-nilpotency of G directly, and in the second case we can effectively construct the trace subshift $\Sigma_1(G)$ of G using Theorem 1.4. Note that G is nilpotent if and only if $\Sigma_1(G)$ is eventually constant, that is, there is state $s \in S$ and number N such that $y_i = s$ for all $y \in \Sigma_1(G)$ and $i > N$. This condition can be easily verified from the finite automaton representation of the trace subshift. ■

4. The case of reversible CA

4.1. Mortality

Analogously to Theorem 3.1, a direct application of the undecidability of the tiling problem among two-way deterministic tile sets was used in [9] to show that global mortality is undecidable among reversible CA:

Theorem 4.1. *It is undecidable whether a given reversible 1D CA is globally mortal. This holds even if the CA is known to be expansive* [9].

Proof. For a given two-way deterministic tile set T we start by adding to it tiles until we have a tile set $S = T \cup F$ that is two-way deterministic and complete in the sense that for every horizontal color h and vertical color v there is a unique tile $t \in S$ with $N(t) = h$ and $W(t) = v$, as well as a unique tile $t' \in S$ with $S(t) = h$ and $E(t) = v$. The added tile set F can be effectively constructed by arbitrarily pairing the missing horizontal/vertical color pairs among NW- and SE-edges in T.

One can next construct a one-dimensional CA over the state set S analogously to the proof of Theorem 3.1: Let s_1 and s_2 be the states of a cell and its right neighbor, respectively. Then the new state of the cell is the unique tile $t \in S$ such that

$$W(t) = E(s_1) \text{ and } N(t) = S(s_2).$$

It follows from two-way determinism of S that the CA is reversible.

The CA is globally mortal with respect to the added tiles F if and only if T does not admit a tiling. Indeed, if T admits a tiling then a diagonal of a valid tiling is a configuration whose space-time diagram only contains elements of T. So the CA is not globally F-mortal.

Conversely, if T does not admit a tiling then there exists a number n such that T does not admit a tiling of an $n \times n$ square. Then every segment of length n will contain a state in F within $2n$ time steps, so the CA is globally mortal.

The first part of the theorem now follows from the undecidability of the tiling problem among two-way deterministic tile sets.

For the second part we observe that global mortality is invariant under composing the CA with the shift: G is globally mortal if and only if $G \circ \sigma^{r+1}$ globally mortal. But $G \circ \sigma^{r+1}$ is always expansive where r denotes the neighborhood radius. ∎

A layer of binary signals was introduced in [9] to obtain the following result.

Theorem 4.2. *It is undecidable whether a given reversible 1D CA is locally mortal. This holds even if the CA is known to be left-expansive* [9].

Proof. We reduce global immortality to local immortality. Let G be an expansive one-dimensional CA, with state set $S = T \cup F$ where $T \cap F = \emptyset$. Let us construct a new CA G' with state set $S \times \{0,1\}^2$. Each state has two binary signals associated with it. The S-states evolve according to G. Normally (if the underlying S-state belongs to T) the two signals travel left and right, respectively. An exception happens in every cell whose state belongs to F: If the incoming right and left moving signals have binary values a and b, respectively, the outgoing right and left moving signals will have binary values $1 + b \pmod 2$ and $a + b \pmod 2$, respectively. Let F' (the new acceptance set) consist of all states that contain a signal with binary value 1.

It is easy to see that G' is left-expansive, and it is locally mortal w.r.t F' if and only if G is globally mortal w.r.t F. The result now follows from Theorem 4.1. ∎

If one executes the constructions in the proofs of Theorems 4.1 and 4.2 starting with an aperiodic, two-way deterministic tile set T then the final CA G' is left-expansive but not regular. Aperiodic two-way deterministic tile sets must exist due to the undecidability of the tiling problem — and an aperiodic four-way deterministic example was even explicitly constructed in [8] — so we have the following corollary:

Corollary 4.3. *There are left-expansive reversible CA that are not regular.*

By adding yet another layer of signals we can reduce local mortality among left expansive CA into the problem of determining if a given CA is left-expansive:

Theorem 4.4. *It is undecidable whether a given reversible 1D CA is left-expansive* [9].

Proof. Let G be a left-expansive CA over the state set $S = T \cup F$ where $T \cap F = \emptyset$. Construct a new CA G' with two additional binary signals in each cell. The first signal shifts left while the second one does not move. In addition, in each cell where the underlying S-state belongs to F the two signals are swapped. It is easy to see that G' is left-expansive if and only if G is locally mortal with respect to F. The result now follows from Theorem 4.2. ■

It is not known whether expansivity of reversible CA is decidable. Note that if locally mortality were undecidable among expansive CA then the construction we used in Theorem 4.4 would show that expansivity is undecidable. Global mortality is known to be undecidable among expansive CA (Theorem 4.1), but it remains a challenge to reduce global mortality to local mortality while preserving expansivity. For left-expansive CA this was successfully done in the proof of Theorem 4.2.

4.2. Equicontinuity

Using a reduction from the mortality problem of reversible Turing machine it is shown in [7] that it is undecidable if a given one-dimensional CA is periodic:

Theorem 4.5. *It is undecidable whether a given reversible 1D CA is periodic* [7].

Using Theorem 1.2(b) we directly obtain the following:

Corollary 4.6. *It is undecidable if a given reversible one-dimensional CA is equicontinuous* [7].

By observing that periodicity of a sofic shift is decidable, one sees analogously to Corollary 3.4 that the classes (L1) and (L3) in the language classification of Section 1.7 are recursively inseparable even among reversible CA.

Corollary 4.7. *It is undecidable if a given reversible one-dimensional CA is regular. More precisely, periodicity and non-regularity are recursively inseparable among reversible one-dimensional CA* [16].

Proof. Suppose there exists an algorithm to separate periodic from non-regular reversible CA. Then we can decide periodicity of given G as follows: The separating algorithm either tells that G is definitely not periodic or that G is definitely regular. In the first case we know the non-periodicity of G directly, and in the second case we can effectively construct the trace $\Sigma_1(G)$ of G using Theorem 1.4. It is an easy matter to effectively determine if $\Sigma_1(G)$ is periodic, given the finite automaton representation of its language. ■

Finally, we mention without proof that recently V.Lukkarila reduced the halting problem of reversible Turing machines to show that sensitivity is undecidable among reversible CA:

Theorem 4.8. *It is undecidable whether a given reversible 1D CA is sensitive* [16].

5. Conclusions and open problems

We have discussed recent undecidability results from [9, 7, 16]. These show that many dynamical properties that were previously known to be undecidable for general one-dimensional CA remain undecidable with the additional constraint of reversibility.

Many challenging and interesting decidability problems remain open. Most notably, it is not known whether it is decidable if a given one-dimensional reversible CA is expansive. Another, closely related open question is a conjecture of Nasu [11] that all expansive one-dimensional cellular automata are conjugate to subshifts of finite type, or equivalently, that the column subshift $\Sigma_{r+1}(G)$ of width $r+1$ is of finite type when G is expansive and r is its neighborhood radius. It would be even interesting to know if all expansive CA are regular, a weaker statement than Nasu's conjecture. Note that Corollary 4.3 states that left-expansive CA are not necessarily regular.

One approach to the expansivity problem is to try to reduce global mortality to local mortality of expansive CA, as suggested in Section 4.1. It would seem, however, that for this to work a counter example to Nasu's conjecture would be needed, in an analogy to Corollary 4.3.

We also have not yet touched the decidability status of any mixing property such as transitivity.

Acknowledgement

The author wishes to thank N.Ollinger and V.Lukkarila for discussions and fruitful collaboration.

References

1. B.Durand, E.Formenti, and G.Varouchas, *On undecidability of equicontinuity classification for cellular automata*, Discrete Models for Complex Systems, DMCS'03 (Michel Morvan and Éric Rémila, eds.), DMTCS Proceedings, vol. AB, Discrete Mathematics and Theoretical Computer Science, 2003, pp. 117–128.
2. F.Blanchard and P.Tisseur, *Some properties of cellular automata with equicontinuity points*, Ann. Inst. Henri Poincar, Probabilits et Statistiques **36** (2000), no. 5, 569–582.
3. G.A.Hedlund, *Endomorphisms and automorphisms of the shift dynamical system*, Mathematical Systems Theory **3** (1969), 320–375.
4. J.Cervelle, E.Formenti, and P.Guillon, *Sofic trace subshift of a cellular automaton*, Computation and Logic in the Real World, CiE'07 (B.Löwe S.B.Cooper and A.Sorbi, eds.), Lecture Notes in Computer Science, vol. 4497, Springer, 2007, pp. 152–161.
5. J.Kari, *The nilpotency problem of one-dimensional cellular automata*, SIAM Journal on Computing **21** (1992), 571–586.
6. ———, *Theory of cellular automata: A survey*, Theoretical Computer Science **334** (2005), 3–33.
7. J.Kari and N.Ollinger, *Periodicity and immortality in reversible computing*, Submitted for publication (`oai:hal.archives-ouvertes.fr:hal-00270815_v1`) (2008).
8. J.Kari and P.Papasoglu, *Deterministic aperiodic tile sets*, Geometric and functional analysis **9** (1999), 353–369.
9. J.Kari and V.Lukkarila, *Some undecidable dynamical properties for one-dimensional reversible cellular automata*, To appear (2008).
10. P. Di Lena, *Decidable properties for regular cellular automata*, Fourth IFIP International Conference on Theoretical Computer Science, TCS 2006 (L.Bertossi G.Navarro and Y.Kohayakawa, eds.), IFIP International Federation for Information Processing, vol. 209, Springer, 2006, pp. 185–196.

11. M.Nasu, *Textile systems for endomorphisms and automorphisms of the shift*, Mem. Amer. Math. Soc., vol. 546, AMS, 1995.
12. P.Kurka, *Languages, equicontinuity and attractors in cellular automata*, Ergodic Theory and Dynamical Systems **17** (1997), 417–433.
13. R.Berger, *The undecidability of the domino problem*, Mem. Amer. Math. Soc. **66** (1966), 1–72.
14. R.H.Gilman, *Notes on cellular automata*, Preprint (1988).
15. V.Lukkarila, *The 4-way deterministic tiling problem is undecidable*, Submitted for publication (2007).
16. ———, *Sensitivity is undecidable for reversible cellular automata*, Under preparation (2008).

ALGORITHMS AND COMPUTATIONS ON CELLULAR AUTOMATA

J. MAZOYER

Université de Lyon,
Laboratoire de L'Informatique et du Parallélisme,
École Normale Supérieure de Lyon, CNRS, Université Claude Bernard,
Institut des Systèmes Complexes.
E-mail address: mazoyer@ens-lyon.fr

Since their origin, algorithms on cellular automata have been defined. The noticeable point is that algorithms and computations are disjoints notions. Besides many computations (in the framework of Turing computability) as multiplication, real-time recognition of primes in unary, ..., algorithms without sequential counterpart have been defined as Firing Squad Synchronization Problem, French Flag, self-reproduction, ...

We recall the notion of signal or particle. It is simply the trajectory (a neighborhood-connected line) of an information (subset of states up to a grouping operation) in the orbit of some configuration. Basic signals are straight lines; from them a large variety of signals may be constructed on a finite configuration: but all signals are not constructible and it remains to characterize (if possible) such signals. When the initial configuration is infinite, all signal are constructible: in some way input (of some computation) may define new signals.

On other hand, a signal viewed as a connected line in an orbit may be constructed in different manners: so, states and their order used to defined a signal may be viewed as data setting on the signal. Many algorithms in a geometrical way move this kind of data. We briefly indicate main algorithms showing how to translate, reverse dat and to carry data from one signal to another one. We also give an example of how transform parts of an orbit using tiling of the orbit respecting the natural dependencies of cellular automata. We observe that there exist in literature a large variety of methods; so, it is quite impossible to list all of them.

Since work of K. Čulik and C. Choffrut, it is well done how to transform the graph of dependencies of a 1D-CA of neighborhood $\{-1, 0, +1\}$ into a graph of dependencies of a 1D-CA of neighborhood $\{-1, 0\}$ and then to see the initial cellular automaton as a trellis automaton. This leads us to the notion a grid (an injection of $\mathbb{Z} \times \mathbb{N}$ into $\mathbb{Z} \times \mathbb{N}$). Given a cellular automaton $\mathcal{A}$ which constructs a grid G on the initial configuration $c_{\mathcal{A}}$ and an another cellular automaton $\mathcal{B}$, it is possible to algorithmically defined a new cellular automaton $\mathcal{C}$ such that from every initial configuration $c_{\mathcal{B}}$, a new configuration $c_{\mathcal{C}}$ has an orbit such that the orbit of $c_{\mathcal{B}}$ appears on grid G. This is a convenient way to move the whole orbit of $\mathcal{B}$ in space-time. Another interest of grids is that when communication time between cells or computational time of cells are not uniform, the grid is modified but remains a grid: this allows, in some sense, to bypass to synchronous character of cellular automata.

We study a typical example of Turing computation: the function $(x, y) \longrightarrow (xy)^2$. This points that often "good" cellular computational algorithms are the "natural" ("human") ones. This example also show how to compute composition of functions ($(x, y) \longrightarrow xy$ and $z \longrightarrow z^2$) using grids. On a grid, we have two basic moves of information (to the right and to the left), to stay on a cell (time move) is now to achieve a right move followed by a left one. To compose functions we put first computation on a grid G_1 and second computation on a grid G_2 the right move of which is a time move of G_1 (we have choice for its left move).

Then we propose some hints to achieve a programming language for 1D-CA:

- The *s-n-m* theorem has two components: the classical one on local (cell based) computations and a global one on the grid constructed during the computation. In particular, not only the shape of the grid but also the size of its holes depend on the data.
- Memory garbage is now synchronization of the active part (computations make several composition and the efficient of the sequence of grids decrease).
- Recursive calls are now simply to construct a new grid in holes of the previous one.
- A **while** instruction is to construct an infinite family of grids.
- Clearly, problems about **for** loops remain.

Finally, we show that the situation for 2d-CA is far more complicated.

Journées Automates Cellulaires 2008 (Uzès), pp. 17-17

AUTOMATA, TILINGS AND TOMOGRAPHY

M. NIVAT

Journées Automates Cellulaires 2008 (Uzès), pp. 18-28

SPARSE SETS

LAURENT BIENVENU [1], ANDREI ROMASHCHENKO [2], AND ALEXANDER SHEN [1]

[1] LIF, Aix-Marseille Université, CNRS, 39 rue Joliot Curie, 13013 Marseille, France
E-mail address: Alexander.Shen@lif.univ-mrs.fr
URL: http://www.lif.univ-mrs.fr/

[2] LIP, ENS Lyon, CNRS, 46 allée d'Italie, 69007 Lyon, France
E-mail address: Andrei.Romashchenko@ens-lyon.fr
URL: http://www.ens-lyon.fr/

Abstract. For a given $p > 0$ we consider sequences that are random with respect to p-Bernoulli distribution and sequences that can be obtained from them by replacing ones by zeros. We call these sequences *sparse* and study their properties. They can be used in the robustness analysis for tilings or computations and in percolation theory.

This talk is a report on an (unfinished) work and is based on the discussions with many people in Lyon and Marseille (B. Durand, P. Gacs, D. Regnault, G. Richard a.o.) and Moscow (at the Kolmogorov seminar: A. Minasyan, M. Raskin, A. Rumyantsev, N. Vereshchagin, D. Hirschfeldt a.o.).

1. Motivation

There are several results which say, intuitively speaking, that "if errors are sparse enough, they do not destroy the intended behavior of the system". For example, percolation theory says that if every edge of a planar conducting grid is cut with probability ε, cuts of different edges are independent and ε is small enough, then the grid remains mostly connected (there is an infinite connected component). Similar statements can be done for cellular automata computations with independent random errors, for tiling with errors etc.

It would be nice to translate these results into the language of algorithmic information theory and define the notion of "individual sparse set" (for a given ε). This notion could be used to split the above mentioned results into two parts: first, we note that with probability 1 with respect to the Bernoulli distribution the resulting set and any its subset is sparse; second, we prove that a sparse set of errors does not destroy the desired behavior (connectivity of the grid, computation of the error-correcting automaton etc.)

2000 ACM Subject Classification: 68Q30, 82B43.
Key words and phrases: algorithmic randomness, percolation theory, error correction.
Thanks to all members and guests of the ESCAPE team at LIF who contributed to the discussions reported in this paper.

Note that the notion of sparse set should take into account not only the overall density of errors but also their distribution. For example, if we cut all the conductors along some line in a grid, we may destroy the connectivity though the fraction of destroyed conductors is negligible (density in a $N \times N$ square tends to 0 as $N \to \infty$).

In this paper we suggest such a definition of p-sparse set and outline some its properties as well as possible applications.

2. Definition of sparse sets

Let $p > 0$ be some rational number. Consider a Bernoulli distribution B_p on the space Ω of all infinite sequences of zeros and ones where bits are independent and each bit equals 1 with probability p. This is a computable distribution on Ω, so the notion of Martin-Löf random sequence with respect to this distribution is well defined (see, e.g., [2, 6]).

Identifying each sequence ω with a subset of $\mathbb{N}$ that consists of all i such that $\omega_i = 1$, we get a notion of a random subset of $\mathbb{N}$ according to this distribution. Let us call *p-sparse* all these sets and all their subsets. The following proposition justifies this definition:

Theorem 2.1. *Let $0 < p_1 < p_2 < 1$ be two rational numbers. Then every p_1-sparse set is p_2-sparse.*

Proof. We need to prove that any p_1-random sequence α can be converted to a p_2-random sequence by replacing some zeros by ones. Informally, we want to replace each 0 by 1 with probability $q = (p_2 - p_1)/(1 - p_1)$; if this replacement is directed by a q-random (with respect to α-oracle) sequence β then the result will be p_2-random. To prove this, we may use van Lambalgen theorem; it says that the pair (α, β) is random with respect to the product distribution, and the replacement result is therefore random with respect to the image distribution, i.e., p_2-Bernoulli distribution. (We do not go into the details since we prove more general result about coupling below.) ∎

Remark. We have defined the notion of a p-sparse set for subsets of $\mathbb{N}$, but the Bernoulli distribution is invariant under permutations and Bernoulli-random sequences remain Bernoulli-random after computable permutations of the domain. Therefore this notion can be defined for subsets of $\mathbb{Z}$, $\mathbb{Z}^k$, sets of strings etc.

By definition, every p-Bernoulli random sequence α is p-sparse. The next theorem shows that there could be another reason of α to be p-sparse:

Theorem 2.2. *For every computable $p \in (0, 1)$ and every p-Bernoulli random sequence α one could replace infinitely many zeros in α by ones but still have p-Bernoulli random sequence* (*for the same* p).

Proof. (Discovered independently by Peter Gacs.) Consider a diverging series with converging squares, say, $q_i = 1/i$. Then consider a sequence β that is random with respect to the distribution of independent bits where ith bit is 1 with probability q_i. Moreover, let us assume that β is random even relative to the oracle α. Then we can check that $\alpha_i \vee \beta_i$ will contain infinitely many additional ones compared to α (since $\sum q_i$ diverges and β is independent of α). On the other hand, $\alpha_i \vee \beta_i$ is still random with respect to p-Bernoulli distribution since its natural distribution $(p + (1 - p)q_i)$ is equivalent to p-Bernoulli distribution due to the effective version of Kakutani's theorem (because the sum of *squares* of the differences between probabilities of the two Bernoulli measures *converges*). ∎

3. Criteria

One would like to have a more straightforward definition for p-sparse sets that does not involves the existential quantifier ("there exists a random sequence such that...''). Such a criterion indeed can be obtained if we restrict ourselves to monotone cylinders in the Martin-Löf definition of randomness.

Let X be a finite subset of $\mathbb{N}$. Consider the set Γ_X of all $\omega \in \Omega$ that have ones at all positions in X. The p-Bernoulli measure of Γ_X equals p^k where k is the cardinality of X.

Let us call a set $N \subset \Omega$ *effectively p-monotone null* set if there exists an algorithm that given rational $\varepsilon > 0$ enumerates a sequence $X_1, X_2, \ldots$ of finite subsets of $\mathbb{N}$ such that the union of corresponding monotone cylinders Γ_{X_i} has measure at most ε (with respect to p-Bernoulli distribution) and covers N.

Theorem 3.1.

A. *For every rational $p > 0$ there exists the maximal effectively p-monotone null set that contains all effectively p-monotone null sets.*

B. *A sequence ω is p-sparse if and only if ω does not belong to the maximal effectively p-monotone null set.*

Proof. **A**. This can be proved in the same way as Martin-Löf theorem about the existence of the maximal effectively null set: one can enumerate all algorithms, modify them to guarantee the bound for the measure and then take the union of all corresponding sets.

B. (See [3].) Let us note that every effectively p-monotone null set N is an effectively null set by definition; moreover, all the sequences obtained from the elements of N by replacing zeros with ones still form an effectively null set. Therefore, the elements of N are not p-sparse.

On the other hand, we have to prove that the set U of all sequences that cannot be made p-random by replacing zeros by ones is an effectively p-monotone null set. By definition, a sequence ω belongs to U if the set E_ω of all sequences that can be obtained from ω by such a replacement is a subset of the maximal effectively null set M (with respect to p-Bernoulli measure). We need (for a given $\varepsilon > 0$) to cover U by the sequence of monotone cylinders of total measure at most ε. Consider the enumerable union of (non-monotone) cylinders that covers M and has total measure at most ε. This union covers the closed (and therefore compact) set E_ω, therefore a finite union of these cylinders also covers E_ω. Those cylinders deal with finitely many positions in E_ω, therefore there exists a monotone cylinder that contains ω and is covered by that union. The set of all monotone cylinders covered by some finite union of (non-monotone) cylinders in the sequence is enumerable, and we get the required enumerable family of monotone cylinders of total measure at most ε. ∎

Remark. This criteria can be used to prove Theorem 2.1: it remains to show that for a monotone set X its p-Bernoulli measure $B_p(X)$ is a monotone function of p. (This is a standard result in classical probability theory that is proved essentially in the same way, by coupling p_1- and p_2-Bernoulli random variables, see below Theorem 4.4.)

It would be interesting to find another equivalent definition of p-sparse sets. One may look for a martingale criterion of sparseness. Recall [4] that a sequence is random if and only if every lower semicomputable martingale is bounded on its prefixes. Trying to characterize sparse sequences, one may try to find the corresponding class of martingales.

Formally speaking, a *p-martingale* (in the algorithmic information theory) is defined as a function $x \mapsto m(x)$ defined on binary strings with non-negative real values such that

$$m(x) = pm(x1) + (1-p)m(x0).$$

If $m(x)$ is interpreted as the capital of the player after bits of x appear during the game (from left to right), this equation says that the game is fair, i.e., the expected capital after the next round equals the capital before it. We say that martingale m *makes bets only on ones* if the outcome 1 is always more profitable than 0, i.e., if

$$m(x0) \le m(x1)$$

for every binary string x. A stronger condition: a martingale m is *monotone* if $m(x) \le m(y)$ when $x \prec y$, i.e., when y can be obtained from x by replacing some zeros by ones. It turns out that monotone martingales can be used for defining sparse sets (sequences) while martingales that bet only on 1's are not suitable (there are too many of them).

Recall that a martingale m is *lower semicomputable* if there exists a computable function $(x, n) \mapsto M(x, n)$ with rational values (here x is a string and n is a natural number) such that for every x the sequence $M(x, 0), M(x, 1), \ldots$ is non-decreasing and converges to $m(x)$.

Theorem 3.2.

A. *If a sequence α is not p-sparse, there exists a monotone lower semicomputable martingale m that tends to infinity on the prefixes of α.*

B. *If there exists a monotone lower semicomputable martingale m that is unbounded on the prefixes of α, then α is not p-sparse.*

C. *There exists a p-sparse sequence α and a lower semicomputable martingale m that makes bets only on ones and still tends to infinity on the prefixes of α.*

Proof. **A**. For every enumerable union T of monotone cylinders we may consider a lower semicomputable martingale m_T by letting $m_T(x)$ be the conditional probability to get a sequence in T starting from x (and adding independent p-Bernoulli bits). Since T is monotone, this martingale is also monotone; it starts from the measure of T and reaches 1 at any sequence in T. Adding up these martingales (say, take T_n of measure at most 4^{-n} and multiply the corresponding martingale by 2^n), we get a lower semicomputable martingale that tends to infinity on the prefixes of all elements of given monotone effectively null set.

B. Let m be any lower semicomputable monotone martingale. For a given c we may consider the enumerable set of all x such that $m(x) > c$, and all (non-monotone) cylinders rooted at these x. The union of these cylinder has measure at most $1/c$ due to martingale inequality. Since the martingale is monotone, we can replace non-monotone cylinders by the monotone ones not changing the union. Doing this for large c, we get the required covering by an enumerable union of monotone cylinders that has small measure.

C. To construct a required counterexample, let us consider a p-Bernoulli random sequence and split its elements into pairs. Then let us modify this sequence replacing pairs 01 and 10 by 00 (and leaving 00 and 11 unchanged). This gives us a p-sparse sequence since we replace ones by zeros in a random sequence. However, the player can make a safe bet on the second bit of each pair if she sees that the first bit is 1, and this happens infinitely many times since we have started from a random sequence. ■

4. Coupling

In this section we review a well known technique that can be then adapted to prove that some operation produce sparse sets.

Let A and B be two finite sets and let $R \subset A \times B$ be a binary relation; we write aRb if $\langle a, b\rangle \in R$. Consider two random variables α and β that range over A and B. defined on unrelated probability spaces. We say that $\alpha R\beta$ (abusing slightly the notation) if there exist random variables α' and β' defined on some common probability space M such that

- α' has the same distribution as α;
- β' has the same distribution as β;
- $\alpha'(m)R\beta'(m)$ for every $m \in M$.

This definition refers not to α and β themselves, but only to the corresponding probability distributions on A and B, so it defines a relation between probability distributions on A and B.

In other terms, we are looking for a matrix of non-negative reals (a distribution on $A \times B$) that has given sums for all rows and columns. This task can be reformulated in terms of network flows. Indeed, consider a bipartite graph that has A on the left and B on the right. The source s is connected with all elements of A by edges whose capacities are given probabilities of the elements in A; all elements of B are connected to the sink t by edges whose capacities are probabilities of the elements of B. The edges between A and B have unlimited capacities and correspond to the elements of R. Then $\alpha R\beta$ means that this network has a flow of size 1. The Ford–Fulkerson theorem provides a criterion for the existence of such a flow; this criterion describes the obstacles for $\alpha R\beta$. A pair of sets $S \subset A$ and $T \subset B$ is an *obstacle* if

- all R-neighbors of all elements in S belong to T
- $\Pr[\alpha \in S] > \Pr[\beta \in T]$

It is evident that if such an obstacle exists, then $\alpha R\beta$ is not possible, and Ford–Fulkerson duality says that this condition is necessary and sufficient:

Theorem 4.1. *$\alpha R\beta$ if and only if there are no obstacles of described type.*

Now we extend this result to infinite sequences. Consider two random variables α and β that range over A^∞ and B^∞ (here X^∞ stands for the set of all sequences $x_0, x_1, \ldots$ where $x_i \in X$). We say that $\alpha R\beta$ if there exist α' and β' that share the same probability space M, have the same distribution as α and β, and $\alpha'_i(m)R\beta_i(m)$ for every $m \in M$ and for every $i \in \mathbb{N}$.

Note that this definition refers only to the distributions on A^∞ and B^∞ and that the relation on $A^\infty \times B^\infty$ is defined coordinate-wise (separately for each i). It turns out that the statement about possible obstacles remains true for this definition.

Theorem 4.2. *The relation $\alpha R\beta$ is false if and only if there exist Borel sets $S \subset A^\infty$ and $T \subset B^\infty$ such that:*

- *for every $a_0a_1 \ldots \in S$ every $b_0b_1 \ldots \in B^\infty$ such that $\forall i\, (a_iRb_i)$ belongs to T;*
- $\Pr[\alpha \in S] > \Pr[\beta \in T]$

Proof. It is evident that the existence of S and T with these properties is indeed an obstacle for $\alpha R\beta$. In the other direction we cannot use just the Ford–Fulkerson argument since the spaces are infinite. However, the relation is defined coordinate-wise, and we may for every N find a joint distribution on $A^N \times B^N$ that has the same projections on A^N and B^N as α

and β, and has the required property with respect to R. (Indeed, an obstacle to this finite task as described in Theorem 4.1 at the same time is an obstacle in our current sense.) It remains to find limit point as $N \to \infty$ using the standard compactness argument. ∎

Note that the existential quantifier for one of the sets S and T can be eliminated: the obstacle can be defined as the set S such that the set of all its neighbors have β-probability less than the α-probability of T. This evident remark shows that defined relation is transitive:

Theorem 4.3. *Let A, B, C be finite sets; let $R_1 \subset A \times B$ and $R_2 \subset B \times C$ be two binary relations and $R = R_1 \circ R_2$ be its composition. Then $\alpha R_1 \beta \wedge \beta R_2 \gamma \;\Rightarrow\; \alpha R \gamma$ for any random variables α, β, γ that range over $A^\infty, B^\infty, C^\infty$ respectively.*

We will mostly deal with the special case where $A = B = \{0, 1\}$ and the relation R is linear order $\leq$. Let us denote the corresponding relation between two random variables α and β that range over $\Omega = \{0, 1\}^\infty$ by $\alpha \preceq \beta$. (The same notation will be used for corresponding distributions.) In this case the description of obstacle can be simplified further:

Theorem 4.4. *$\alpha \preceq \beta$ if and only if* $\Pr[\alpha \in Z] \leq \Pr[\beta \in Z]$ *for every monotone Borel set $Z \subset \Omega$.*

(A set $Z \subset \Omega$ is monotone if $\alpha \in Z$ and $\forall i\, (\alpha_i \leq \beta_i)$ implies $\beta \in Z$.)

Proof. One could argue that the set $T(S)$ of neighbors (as defined above) for every set $S \subset \Omega$ is monotone and contains S, so S can be replaced by $T(S)$. (Note that $T(T(S)) = T(S)$.)

However, there is a technical problem since we need T to be a Borel set. This can be avoided if we use this argument for finite case $\{0, 1\}^N$ and take a limit point after that. ∎

Remark. Similar statements can be made for any finite set and partial preordering on it (instead of the standard ordering of $\{0, 1\}$).

5. Coupling and algorithmic randomness

We want to apply coupling technique to establish results about random and sparse sets. The following two statements will be used as main technical tools for this.

Let $\mathcal{M}$ be an oracle machine that accepts or rejects its input (natural number) n asking questions of type "$m \in L$?" for different natural numbers m and oracle L. Such a machine defines a mapping that maps the oracle set L into the set accepted by $\mathcal{M}$ when oracle L is used.

More precisely, this is a partial map, since it may happen that for some oracles L the machine $\mathcal{M}^L$ does not terminate for some inputs. We are not interested in the partial functions and consider $\mathcal{M}(L)$ as undefined in these cases. Recall also that we identify sets $X \subset \mathbb{N}$ and their characteristic sequences, so an oracle machine defines a (partial) mapping $\Omega \to \Omega$.

Let P be a computable probability distribution on Ω and let $\mathcal{M}$ be an oracle machine. For a random variable α that ranges over Ω we consider the image distribution $\mathcal{M}(P)$ on Ω, i.e., the distribution for the variable $\mathcal{M}(\alpha)$. This distribution is well defined if $\mathcal{M}(\omega)$ is defined with probability 1. We assume that this is the case; then $Q = \mathcal{M}(P)$ is a computable distribution on Ω. The following natural connection between P-randomness and Q-randomness [5] holds:

Theorem 5.1.

A. *If ω is Martin-Löf P-random sequence, then $\mathcal{M}(\omega)$ is Martin-Löf Q-random sequence.*

B. *Any Martin-Löf Q-random sequence τ equals $\mathcal{M}(\omega)$ for some Martin-Löf P-random sequence ω.*

Proof. **A**. If ω is Martin-Löf P-random and $\mathcal{M}(\omega)$ is defined, this is a direct corollary of the definitions: if Z is a Q-effectively null set containing $\mathcal{M}(\omega)$ then its preimage $\mathcal{M}^{-1}(Z)$ is a P-effectively null set containing ω (it is an effectively null set since the preimage of an effectively open set of Q-measure less than ε is an effectively open set of the same P-measure).

However, there is one more thing to prove: we need to show that $\mathcal{M}(\omega)$ is defined on every P-random ω. Indeed, for every n the set of all oracles ω such that $\mathcal{M}^{\omega}$ is defined on n is an effectively open set of full measure. It is easy to see that its complement is an effectively null set and therefore cannot contain a Martin-Löf random sequence.

B. Let N be the maximal P-effectively null subset of Ω. We need to prove that every Q-random sequence has a $\mathcal{M}$-preimage outside N. In other terms, we have to prove that the set N' of sequences that do not have preimages outside N is a Q-effectively null set.

Assume that some rational $\varepsilon > 0$ is given. We have to find an effectively open set of Q-measure less than ε that covers N'. First, let us consider an effectively open set N_ε that covers N and has P-measure less than ε. We want to show that the set of sequences that have no preimages outside N_ε is an effectively open set. (Note that Q-measure of this set is less than ε since Q is the image of P.)

Indeed, consider a sequence ω that has no preimages outside N_ε. As we have seen, $\mathcal{M}$ is defined everywhere outside N (and therefore outside N_ε). So for every $\tau \notin N_\varepsilon$ the infinite sequence $\mathcal{M}(\tau)$ deviates from ω at some place. The set of all τ for which this happens at ith place is open, and these sets together with the open set N_ε form a covering of Ω. Compactness allows to replace this covering by its finite part, and this gives some neighborhood of ω; all elements of this neighborhood have no preimages outside N_ε. Moreover, we can search for all finite coverings of this type, so the set of all sequences that have no preimage outside N_ε is effectively open (and has Q-measure less than ε as we have discussed).

This finishes the argument (which is an effective version of a classical argument proving that if a continuous function is defined everywhere on a compact set, then the image of this set is also compact). ∎

This statement has a simple intuitive meaning: if we have a source of random bits composed of a P-random bit source and an oracle machine $\mathcal{M}$ using those bits as an oracle, what sequences should we expect at the output? There are two possible answers. First, we can say that the internal P-random source can produce any P-random sequence, so we can get images of those sequences at the output. On the other hand, we can ignore the internal structure and say that we altogether have a random bits generator with distribution Q, so Q-random sequences are expected at its output. Theorem 5.1 says that these two classes coincide.

Remark. This question is more difficult (and is not solved yet, as far as we know) if the machine $\mathcal{M}$ is undefined with positive probability. Then we get only a semimeasure as output distribution; one would like to prove that the image of the set of P-random sequences is still determined by this semimeasure, but it is not clear how to prove this and whether it is possible.

The second tool connects coupling with algorithmic randomness.

Theorem 5.2. *Let P and Q be two computable distributions such that $P \preceq Q$ and, moreover, there exists a computable distribution on $\Omega \times \Omega$ that has projections P and Q and $\alpha_i \le \beta_i$ has probability one according to this distribution for every i.*

Then every P-random sequence can be transformed into a Q-random sequence by replacing some zeros by ones. Similarly, every Q-random sequence can be transformed into a P-random one be replacing some ones by zeros.

Proof. It would be nice to get rid of the additional condition: the computability of the distribution on pairs. But with this condition (that is easy to check in all applications below) the statement of the theorem is a direct consequence of Theorem 5.1. Using projection as $\mathcal{M}$, we see that every P-random sequence is a first term of a random pair, and the second term of this pair is a Q-random sequence we looked for. (The same argument can be used for choosing a P-random sequence for a given Q-random one.) ∎

Note that in this way we get one more proof of Theorem 2.1. More interesting applications will be given in the next section.

6. Operations that preserve sparseness

In this section we want to prove that some transformations preserve sparseness (though may change the value of parameter p). Let us start with a simple example.

Theorem 6.1. *Let $a_0, a_1, a_2, \ldots$ be a p-sparse sequence. Then the sequence*

$$a_0, a_0, a_1, a_1, a_2, a_2, \ldots$$

(each bit is doubled) is $\sqrt{p}$-sparse.

Proof. Since bit doubling is a monotone transformation, we may assume that $a_0, a_1, \ldots$ is a random sequence with respect to the Bernoulli distribution B_p. Theorem 5.1 says that the sequence $a_0, a_0, a_1, a_1, a_2, a_2, \ldots$ is then random with respect to the image distribution $D(B_p)$ where D doubles each bit.

Theorem 5.2 shows that it remains to prove that $D(B_p) \preceq B_{\sqrt{p}}$. Since both distributions are products of (independent) distributions on bit pairs, it is enough to consider these distributions on pairs. They have the same probability of 11 combination (p in both cases) while 01 and 10 have zero probability in $D(B_p)$ and positive probability $\sqrt{p}(1-\sqrt{p})$ for $B_{\sqrt{p}}$. So we can start with $B_{\sqrt{p}}$ distribution and then replace 1 by 0 if the other bit is 0. ∎

More complicated tools are needed in the other example.

Theorem 6.2. *Let $a_0, a_1, a_2, \ldots$ be a p-sparse sequence. Then the sequence*

$$a_0 \vee a_1, a_1 \vee a_2, a_2 \vee a_3, \ldots$$

is $2\sqrt{p}$-sparse.

Note that this sequence is an upper bound for $a_1, a_1, a_3, a_3, a_5, a_5, \ldots$, so this result implies that the latter sequence is also sparse (though for a larger value of p compared with Theorem 6.1).

Proof. The sequence in question can be represented as bitwise OR of two sequences:

$$\begin{array}{ccccccccc}
a_0 & a_2 & a_2 & a_4 & a_4 & a_6 & a_6 & a_8 & \ldots \\
a_1 & a_1 & a_3 & a_3 & a_5 & a_5 & a_7 & a_7 & \ldots \\
a_0{\vee}a_1 & a_1{\vee}a_2 & a_2{\vee}a_3 & a_3{\vee}a_4 & a_4{\vee}a_5 & a_5{\vee}a_6 & a_6{\vee}a_7 & a_7{\vee}a_8 & \ldots
\end{array}$$

If $a_0, a_1, a_2, \ldots$ is a random sequence with Bernoulli distribution, then the first two sequences are independent and their distributions (as we have seen above) are $\preceq$-below $B_{\sqrt{p}}$. It follows easily that the disjuction of these two sequences is $\preceq$-below the disjuction of two Bernoulli distributions $B_{\sqrt{p}}$ and therefore is below $B_{2\sqrt{p}}$-distribution. It remains to apply Theorem 5.2 (the computability condition is easy to check). ∎

Similar argument can be applied to the sequence

$$a_0 \vee a_1 \vee \ldots \vee a_{k-1}, a_1 \vee a_2 \vee \ldots \vee a_k, \ldots$$

for any k and shows that it is $k\sqrt[k]{p}$-sparse if the initial sequence is p-sparse. In terms of sets we get the following result (note that d-neighborhood of a point in $\mathbb{N}$ or $\mathbb{Z}$ consists of $2d+1$ points):

Theorem 6.3. *For every positive integer $d > 0$ and every p-sparse set A the d-neighborhood of A is q-sparse for $q = (2d+1)\sqrt[2d+1]{p}$.*

For simplicity we may ignore the exact value of the probability and say something like "the neighborhood of a sparse set is sparse". The exact meaning of this claim is that for every $q > 0$ there exists $p > 0$ such that the neighborhood of every p-sparse set is q-sparse. This is true for subsets of $\mathbb{Z}^2$ or $\mathbb{Z}^k$ (the proof works for L_∞-neighborhoods as before; then the statement can be extended to any type of neighborhood).

One would like to strengthen this theorem and prove that the union of two p-sparse sets is q-sparse if p is much less than q. Unfortunately, this cannot be done:

Theorem 6.4. *For every $p, q \in (0,1)$ there exist two p-sparse sets whose union is not q-sparse.*

Proof. (Discovered independently by Denis Hirschfeldt.) Note that for every $q \in (0,1)$ (even very close to 1) and every N (even very large) every q-sparse sequence α splitted into N-bit blocks must contain a block of N zeros.

On the other hand, for every p for large enough N there are two p-sparse sequences α and β whose union $\alpha \vee \beta$ does not have this property. If $p = 1/2$, it is trivial: take $N = 1$ and two complementary $1/2$-random sequences.

In the general case, take N large enough so that $(1-p)^N < 1/2$. Then two events (for a fixed N-bit block) "all N α-bits are zeros" and "all N β-bits are zeros" can be made disjoint keeping the p-Bernoulli distributions for α and β in the block unchanged. Making all blocks independent, we get a computable distribution on pair of sequences that has p-Bernoulli projections (marginal distributions) and is concentrated on pairs of sequences that have the required property. ∎

Another source of sparse sequences is provided by the following theorem:

Theorem 6.5. *Let P be a computable distribution on Ω and let $\alpha = a_0a_1a_2\ldots$ be a random sequence with respect to this distribution. If all the conditional probabilites of ones along this path (i.e., $\Pr[x_i = 1 | x_0 \ldots x_{i-1} = a_0 \ldots a_{i-1}]$) do not exceed some rational p, then α is p-sparse.*

Proof. Consider the following randomized machine. It uses uniform Bernoulli distribution to generate a sequence of independent random variables $\xi_0, \xi_1, \xi_2, \ldots$ distributed uniformly in $[0, 1]$ (by splitting random bits into countably many groups). Then these random variables are used to produce bits that have distribution P: we compare ξ_0 with the probability of 1 at the first place to get a_0, then we compare ξ_1 with the (conditional) probability of 1 after a_0 to get a_1, etc.

Theorem 5.1 guarantees that in this way we can obtain exactly P-random sequences. Replacing all thresholds by p, we in parallel get p-Bernoulli random sequence that guarantees that the sequence $a_0a_1\ldots$ is p-sparse.

Note a subtle point in this argument: we do not claim that the $P \preceq B_p$ since the inequality for conditional probabilities is guaranteed only along the path $a_0a_1a_2\ldots$; however, the sequence generated in parallel with $a_0a_1a_2\ldots$ is p-random since the image of every random sequence $\xi_0\xi_1\ldots$ is p-random. ∎

7. Using sparse sets in error analysis

Now several results about robustness may be reformulated in terms of sparse sets. Let us give an example from percolation theory.

Consider a grid of vertical and horizontal lines (as in the cell paper) that splits the plane into unit squares. Each node (line crossing) is a contact, and each edge (of a unit square) is a conductor.

Percolation theory says that if each conductor is independently cut with sufficiently small probability, the network remains essentially connected (has an infinite connected component) with probability 1. Now this can be reformulated in terms of sparse sets:

Theorem 7.1. *If the set of deleted edges is p-sparse for small enough p, the remaining network has an infinite connected component.*

Proof. It is enough to show this for the case of p-random network failures (since the property of having an infinite connected component is monotone).

Therefore, we need to show that the set of measure zero (of all networks that have no infinite connected components or have more than one connected component) is in fact an effectively null set. This can be done by analyzing one of the classical proofs of this result (we omit the details since this is another story). ∎

Now we can apply our results about sparse sets to derive the similar result for node failures (instead of edge failures): failure of a node is equivalent to cutting all four edges that are adjacent to this node.

Theorem 7.2. *If the set of deleted nodes is p-sparse for small enough p, the remaining network has an infinite connected component.*

Proof. It is easy to see that the set of edges adjacent to deleted nodes can be covered by a neighborhood of fixed size of the set of deleted nodes and therefore is also sparse. (Technically, we have twice more edges than nodes, so some technical changes are needed.) ∎

The notion of sparse set can be used in the analysis of tilings that are robust to errors: the results of [1] can be expressed in terms of sparse error sets.

8. Questions

There are some open (as far as we know) questions related to the topic of this paper.

1. Is it possible to replace in the criterion of sparseness the measure of the union of the monotone cylinders by the sum of their measures?

2. One would expect there exists some criterion of p-sparseness in terms of complexity or randomness deficiency. How to define "sparseness deficiency"? One possibility is to take minimum of randomness deficiency (with respect to p-Bernoulli distribution) over all strings obtained from a given one by $0 \to 1$ replacement. Another natural option is to consider lower semicomputable deficiency functions $d_n(x)$ defined on n-bit strings (uniformly in n) that are monotone (i.e., replacement $0 \to 1$ may only increase the deficiency) such that $2^{d_n(x)}$ has average at most 1 (over all n-bit strings).

Are these two definitions connected? close to each other? Can any of them be used to characterize p-sparse sets?

3. Can one get rid of the computability condition in the statement of Theorem 5.2?

4. What can be said about two computable measures P and Q if every P-random sequence can be made Q-random by replacing some zeros with ones? We cannot expect $P \preceq Q$ since this may be not the case even for equivalent measures P and Q (that have the same set of random sequences), but can we claim something weaker in this direction (e.g., P is equivalent to $P' \preceq Q$ or something like this)?

5. What can be said about sets that are p-sparse for every $p > 0$? Can we eliminate the universal quantifier ("for every p") in the definition?

References

[1] Durand B., Romashchenko A., Shen A., *Fixed point and aperiodic tilings*, arXiv:0802.2432v2 (19 Feb. 2008)

[2] Li M., Vitányi P., *An Introduction to Kolmogorov Complexity and Its Applications*, Second Edition, Springer, 1997. (638 pp.)

[3] A. Minasyan, manuscript, 2007 (the course project at the Moscow State University).

[4] Schnorr C.P., *Zufälligkeit und Wahrscheinlichkeit. Eine algorithmische Begründung der Wahrscheinlichkeitstheorie*, Lecture notes in mathematics, v. 218. IV+212 S. Springer, 1971.

[5] Shen A., One more definition of random sequence with respect to computable measure. *First World Congress of the Bernoulli Society on Math. Statictics and Probability theory*, Tashkent, USSR, 1986

[6] Uspensky V.A., Semenov A.L., Shen A., Can a single sequence of zeros and ones be random? *Uspekhi matem. nauk*, 1990, 45, 1, p. 105–162. (Russian; for English translation see *Russian Math. Surveys*, 45:1 (1990), 121–189.; MR 91f:03043.)

Journées Automates Cellulaires 2008 (Uzès), pp. 29-39

TILINGS AND MODEL THEORY

ALEXIS BALLIER [1] AND EMMANUEL JEANDEL [1]

E-mail address: Alexis.Ballier@lif.univ-mrs.fr

E-mail address: Emmanuel.Jeandel@lif.univ-mrs.fr

[1] Laboratoire d'informatique fondamentale de Marseille (LIF), Aix-Marseille Université, CNRS, 39 rue Joliot-Curie, 13 453 Marseille Cedex 13, France

Abstract. In this paper we emphasize the links between model theory and tilings. More precisely, after giving the definitions of what tilings are, we give a natural way to have an interpretation of the tiling rules in first order logics. This opens the way to map some model theoretical properties onto some properties of sets of tilings, or tilings themselves.

1. Introduction

Tilings are a basic and intuitive way to express geometrical constraints; they happened to be of broad interest in computer science since Berger proved the undecidability of the domino problem [2] by showing that they can embed, despite being static objects, some kind of computation. This also was the first step in the links between logics and tilings as they helped to prove the undecidability of some classes for formulae [5, 14, 12, 13]. Some more links have then been discovered by Makowsky that used previous constructions of aperiodic tilesets to show the existence of a complete, finetely axiomatizable and superstable theory [9]. Some recent results by Oger generalize this approach to more abstract definitions of tilings and proves some nice equivalences between model theory and this generalized definition [10].

In this paper we will give details of constructions used to translate tilings and tileset properties into model theoretic ones. Section 2 will be devoted to the proper definitions of tilings and tilesets; We will then translate these definitions into first order formulae in Section 3. Finally in Section 4 we shall present the equivalence results that can be obtained by this translation.

Most of these results are already present in [9, 11]. However we hope that this paper will offer a new look at these results.

The major part of this paper is devoted to tilings of the plane $\mathbb{Z}^2$. However, we may define similar theories for tilings of other spaces such as $\mathbb{Z}^3$ or any Cayley graph. The article [10] in particular deals with tilings of $\mathbb{R}^n$ by polytopes.

2. Tilings

Several definitions of discrete tilings can be found in the litterature, but are equivalent for many purposes [3]. We will focus here on the definition by forbidden patterns.

First we have to define the space we are going to tile: we want to assign a state taken in a finite set Q to each cell of the discrete plane $\mathbb{Z}^2$. Q may be seen as a set of colors, or a set of states. Therefore, we define the set of configurations as the functions from $\mathbb{Z}^2$ to Q:

Definition 2.1. The set of configurations is $Q^{\mathbb{Z}^2}$.

The patterns are nothing but a configuration restricted to a finite domain; that is, considering a finite subset D of $\mathbb{Z}^2$, a pattern is a function from D to Q.

Definition 2.2. A pattern defined on a finite subset D of $\mathbb{Z}^2$ is an element of Q^D.

Informally a tileset represents geometric constraints imposed to the configurations, that is how the states in the cells of the plane are constrained by their neighborhood and how they constrain it. Formally we will define a valid tiling as a configuration that contain no forbidden pattern:

Definition 2.3. A tileset is defined by a finite set of forbidden patterns $\mathcal{F}_\tau$.

A configuration c contains a pattern P defined on D (or equivalently P appears in c) if there exists $x \in \mathbb{Z}^2$ such that:

$$\forall y \in D, c(x + y) = P(y)$$

A configuration is said to be a valid tiling by τ if it contains no pattern in $\mathcal{F}_\tau$.

The so-called domino problem [2] is to know given a tileset whether it generates a valid tiling. The problem has been proven undecidable by Berger in [2].

We will now define a preorder $\preceq$ on configurations that focuses on patterns contained in them. This preorder has been defined in [4, 1], however references to the concept can be found as early as [11]:

Definition 2.4 (The pre-order $\preceq$)**.** Let x, y be two configurations, we say that $x \preceq y$ if any pattern that appears in x also appears in y.

This induces the notion of local isomorphism between two configurations:

Definition 2.5 (Local isomorphism)**.** Two configurations x and y are said to be locally isomorphic if $x \preceq y$ and $y \preceq x$. That is x and y contain the same patterns. We denote it by $x \approx y$.

Two configurations that are equal up to shift are locally isomorphic but the converse is not always true: there exists configurations that are locally isomorphic but one is not a shifted form of the other.

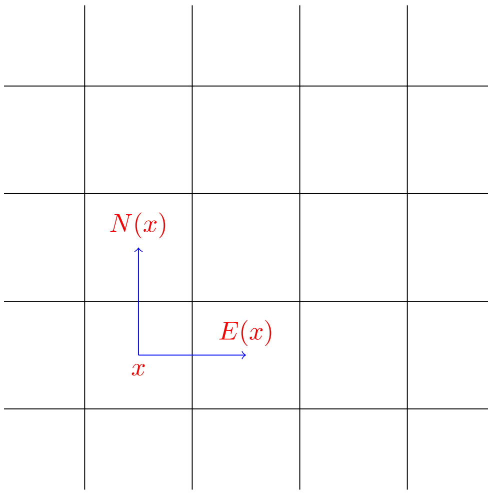


Figure 1: The model we would like to obtain

3. From tilesets to model theory

In this section we translate the definitions given in Section 2 into first order formulae on some given language. This translation maps some properties of tilings onto some other properties of first order logics.

Such a correspondence between tilesets and first order logic has already been defined [11, 9] to show an example of finitely axiomatizable and superstable theory. A similar approach (see 3.4) has been used to prove the undecidability of certain classes of formulae [13, 14, 12, 5].

3.1. Axiomatizing the plane

The ideal model we would like to obtain is the plane $\mathbb{Z}^2$ like depicted in Figure 1. The natural way to define cells on the plane $\mathbb{Z}^2$ is to consider them as variables and the adjacency relations between them as functions that allow us to move north, south, east or west from a given cell:

Definition 3.1. We consider the language with the unary functions for movements on the plane: $\mathcal{L}_0$ is a set of unary functions : $\mathcal{L}_0 = \{N, S, E, W\}$.

And the corresponding axioms of the plane $\mathbb{Z}^2$:

- $\forall x, N(S(x)) = S(N(x)) = E(W(x)) = W(E(x)) = x$
- $\forall x, N(E(x)) = E(N(x))$

These formulae tends to axiomatize $\mathbb{Z}^2$ as a Cayley graph with two generators, the first formula axiomatizing the invertibility of the movements and the second the commutativity. However, these axioms are not sufficient, as we will see in the following sections.

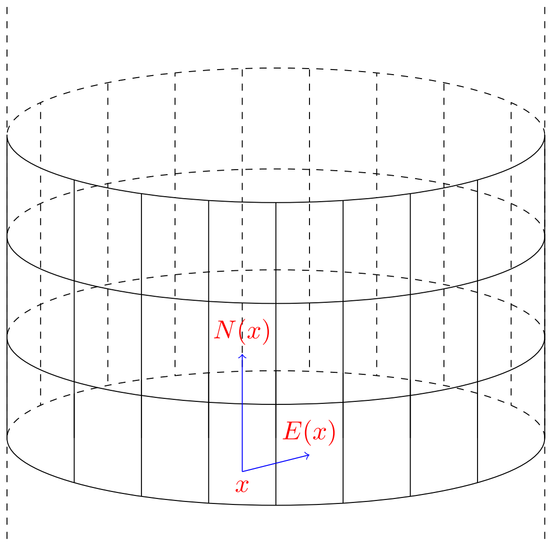


Figure 2: A cylindric model

3.1.1. *Non standard models.* With the axioms of the plane from the previous sections it is still possible to obtain some weird models. First, they also axiomatize some finite models like $\mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}$, or some cylindric models like $\mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}$ (like e.g., in Figure 2).

This problem can be dealt with by adding more axioms : For any i and j we may add the axiom $\forall x, E^i N^j(x) \neq x$. The main problem is then that the number of axioms is not finite, so that (we can prove that) the theory we obtain is not finitely axiomatisable anymore. However in most cases, the presence of these models is not a problem as we can "unfold" them into a plane (see e.g proof of lemma 4.2).

3.1.2. *Connectedness.* The main problem however, which cannot be avoided, is that there is no way to ensure that all models of our theory are connected : A model is said to be connected if any two points can be connected using the N, S, E, W functions. An example of a disconnected model of our theory is depicted on Figure 3. These disconnected models cannot be avoided. This is e.g., a consequence of the Löwenheim-Skolem theorem (There exist models of our theory of arbitrary infinite cardinals, these models cannot be connected if they are not countable) or more simply can be proven by a simple compactness argument: Consider a theory T that axiomatises the plane $\mathbb{Z}^2$. Add two constants c, d and the formulae ϕ_n that express that the points c and d are at distance at least n. Consider the theory $T' = T \cup \{\phi_n \mid n \in \mathbb{N}\}$. Any arbitary finite part of T' admits $\mathbb{Z}^2$ as a model (choose two points c and d arbitrary far) so that T' itself has a model by compactness. Such a model cannot be connected.

This proof also hints to a way to partially solve the problem. Consider formulae $\phi_n(x, y)$ that express that the points x and y are at distance at most n. Now consider the collection $p(x, y) = \{\phi_n(x, y), n \in \mathbb{N}\}$. $p(x, y)$ is a type, that is we can find for every finite part $q(x, y)$ of $p(x, y)$ some points c and d in any model so that $q(c, d)$ is true. Now we are interested

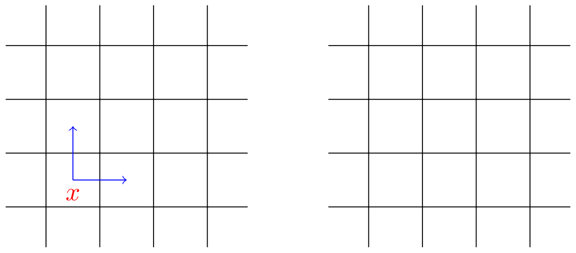


Figure 3: An example of disconnected model

in those models where p is not satisfied, that is in models where there do not exists c and d such that $p(c,d)$ is true. We say that such a model *omits* p

A part of model theory is devoted to the study of models omitting types. As an example, the omitting type theorem states that given a theory T any reasonable type can be omitted. However, most of the classical results in model theory will not work in this context, as e.g. the compactness theorem.

3.2. Encoding configurations

Now that we have some kind of axioms for the plane $\mathbb{Z}^2$, we may define what a configuration is. We defined a configuration as an application from $\mathbb{Z}^2$ to a finite set of states Q. We can code the states of the cells in our theory by unary predicates : we take one predicate Q_i for each state. The only thing we need to ensure is that each cell has exactly one state:

Definition 3.2. New language:

$$\mathcal{L}_Q = \mathcal{L}_0 \cup \{Q_1, \dots, Q_n\}$$

New axioms:

$$A : \forall x, \bigvee_i Q_i(x)$$

$$B : \forall x, \bigwedge_{j \neq i} (Q_i(x) \Rightarrow \neg Q_j(x))$$

We can also reduce the number of predicates by coding the states in binary form: for example, with 4 predicates, we can code up to 16 states.

3.3. The theory of a tileset

Following our definitions of tilesets in Section 2, all what we need to do in order to encode them in first order logic is to write formulae that express ”some specific pattern never appears”. It can be done in the following way : Given a pattern P of domain D, any point p in D can be represented by a function that is a composition of the functions N, S, E, W. We can then write formulae that express that P appears at a point x:

Definition 3.3. A formula to express that a pattern P defined on D appears at point x
$\varphi_P(x) := \bigwedge_{(i,j) \in D} P(i,j)(E^i(N^j(x)))$

As an example, the formula $\varphi = Q_1(E(x)) \wedge Q_2(x)$ expresses that x is of color 2 and its east neighbour is of color 1.

Then the formula $\forall x, \neg\varphi_P(x)$ axiomatizes that P never appears.

Definition 3.4. The theory T_τ of a tileset τ is the theory over the langage $\mathcal{L}_Q$ that contains all previous formulae and the formula $\forall x, \neg\varphi_P(x)$ for each forbidden pattern P. If the set of forbidden pattern $\mathcal{F}_\tau$ is finite, this theory is finitely axiomatisable.

3.4. Other languages

Before proceeding to the results, we give in this section various other languages in which to express tilings.

Another way to represent tilings is with a single unary function s (that intuitively denotes the successor of an integer) and with binary predicates P_i. $P_i(x, y)$ means that the state in the cell (x, y) is i. A structure is then over $\mathbb{Z}$ rather than $\mathbb{Z}^2$.

It is easy to represent forbidden patterns in this language. As an example, the formula $\phi = \forall x, y, \neg\left(P_1(x, y) \wedge P_2(s(x), y)\right)$ means that there cannot be a cell in state 1 at the left of a cell in state 2.

Now suppose that the set of forbidden patterns has some particular form, that is constraints only concern adjacent cells. We now have a set of horizontal constraints H ($(i, j) \in H$ if a cell in state i cannot be at the left of a cell in state j) and vertical constraints V.

Now, the constraints can be written in the following way:

$$\phi = \forall x \forall y \bigwedge_{(i,j)\in H} (P_i(x, y) \Rightarrow \neg P_j(s(x), y)) \wedge \bigwedge_{(i,j)\in V} (P_i(x, y) \Rightarrow \neg P_j(x, s(y))$$

This can be rewritten (by a slight change of variables in the second part of the formula):

$$\forall x \forall y \bigwedge_{(i,j)\in H} (P_i(x, y) \Rightarrow \neg P_j(s(x), y)) \wedge \bigwedge_{(i,j)\in V} (P_i(y, x) \Rightarrow \neg P_j(y, s(x))$$

Now by a straightforward application of the skolemization process, we can replace the function s by a quantifier :

$$\forall x \exists x' \forall y \bigwedge_{(i,j)\in H} (P_i(x, y) \Rightarrow \neg P_j(x', y)) \wedge \bigwedge_{(i,j)\in V} (P_i(y, x) \Rightarrow \neg P_j(y, x')$$

We then obtain a new formula ϕ such that ϕ has a model if and only if there exists a tiling of the plane by the tileset. The proof proceeds as in lemma 4.2 below. Note that the unfolding gives us only a tiling of a quarter of the plane. But it is known that a tileset can tile the entire plane if and only if it can tile a quarter of the plane.

The new formula ϕ is a formula with only three quantifiers $\forall\exists\forall$ and which contains only binary predicates. Thus we actually have proven that the class of formulae $[\forall\exists\forall, (0, \omega)]$ is undecidable. This is the core of the works by Wang, Kahr, Büchi about decidability of class of formulae. We then can deduce by an intricate transformation that the Kahr class $[\forall\exists\forall, (\omega, 1)]$ (one binary predicate, a finite number of unary predicates) is also undecidable.

See [14, 12, 13] for more details. The encoding also has another property : The formula ϕ has a finite model if and only if there exists a periodic tiling of the plane by the tileset. This actually proves that the class $[\forall\exists\forall, (0, \omega)]$ is a *conservative reduction class*. See [5] for more details.

4. Translating tilesets and tilings properties in model theoretical ones

We now show the links between those two different approaches.

Lemma 4.1. *A configuration can be seen as a structure over $\mathcal{L}_Q$. A tiling by τ can be seen as a model of T_τ.*

This lemma is a consequence of the definitions we have taken, any configuration is a structure over $\mathcal{L}_Q$ and the construction of T_τ was done in order to forbid patterns that are forbidden by τ, thus a tiling by τ is a model of T_τ.

Lemma 4.2. *T_τ is consistent if and only if τ can tile the plane.*

Proof. It is obvious (by lemma 4.1) that if τ can tile the plane, then T_τ is consistent: A tiling provides a model of T_τ.

Now suppose that T_τ has a model M. We will "unfold" M starting from a point x in it by applying the functions N, S, E, W that will give us any point in $\mathbb{Z}^2$. We can define a configuration c, such that $c(0,0)$ has the "state" of x, and $c(i,j)$ has the "state" of $E^i(N^j((x)))$. This configuration is a tiling : As M is a model of T_τ, no forbidden pattern can appear. Therefore, from any model of T_τ, we can obtain a tiling of the plane by τ, which finishes the proof. ∎

Remark 4.3. T_τ has a model if and only if T_τ has an infinite model.

We can force all models to be infinite by adding (infinitely many) axioms that will ensure this property. The theory may however not be finitely axiomatisable anymore.

Note however that if a tileset does not admit any periodic tiling, no finite models can appear. Moreover, if a tileset does not admit any tiling with at least one direction of periodicity, then all models are only union of copies of $\mathbb{Z}^2$. That is, no degenerate torus or cylinder may appear.

Lemma 4.4. *T_τ has a finite model if and only if τ can tile periodically the plane.*

Proof. Consider a periodic tiling of period p, we "fold" it into $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/p\mathbb{Z}$ and obtain a model of T_τ since the cell at position $(x+p, y)$ will have the same state as the one at (x, y) or $(x, y+p)$.

If we have a finite model, we unfold it the same way as in Lemma 4.2. It is easy to see that we obtain this way a periodic tiling. ∎

Most of these results can be generalized to tilings of $\mathbb{R}^2$ using "patches" as tiles and we still get the same translation from tileset and tilings into model theory [10].

4.1. Isomorphism

One of the first properties of models of a given theory one has to study is the isomorphism of models. The translation of this property as properties of tilings is quite straightforward:

Lemma 4.5. *Two configurations are equal up to shift if and only if they are isomorphic as structures on the language $\mathcal{L}_Q$.*

Proof. $\Rightarrow$: Let x, y be two configurations equal up to shift and σ be a shift of vector (i, j) such that $x = \sigma(y)$. Then σ is an isomorphism from x to y.

$\Leftarrow$: Let Θ be the isomorphism and a and b two points of x and y such that $\Theta(a) = b$. Then $E^i N^j(a)$ has the same state as $E^i N^j(b)$, as the predicate $P_q(E^i N^j(x))$ has the same value in a and b since Θ is an isomorphism. ■

4.2. Elementary equivalence

Another model theoretic property that translates to tilings is the elementary equivalence. We recall that two structures are elementary equivalent if and only if they satisfy the same formulae (that is have the same theory)

Lemma 4.6 ([11, 10])**.** *Two configurations x and y are locally isomorphic if and only if they are elementary equivalent as structures over $\mathcal{L}_Q$.*

Proof. We will consider for the proof x and y as structures over the language without functions, i.e., we replace in $\mathcal{L}_Q$ the functions N, S, E, W by functional predicates N', S', E', W', that is $N'(x, y) \Leftrightarrow N(x) = y$.

$\Leftarrow$: One can express the apparition of the pattern M by a first order formula like in Definition 3.3: $\exists x, \varphi_M(x)$. Therefore, as any formula valid in one structure is valid in the other one, any pattern that appears in one tiling appears in the other one. This proves that if the structures are elementary equivalent then the tilings are locally isomorphic.

$\Rightarrow$: This proof is rather technical and is given in [10] using Hanf locality lemma [6](lemma 2.3). Hanf locality lemma states that for two structures, if the spheres (using the relational distance) all contain finitely many points (what is always true in our case), and if both stuctures have either the same finite number of different spheres or both have an infinite number, then the two structures are elementary equivalent. Hanf locality lemma can be proved using a back and forth method, or an Ehrenfeucht-Fraïssé game.

In our case, the spheres represent the patterns: Consider a point x and all the points at relational distance at most n, since our language contains only binary predicates and that they represent the functions N, S, E, W, the relational distance is nothing but the L_1 distance (or Manhattan distance or also Taxicab Metric) on $\mathbb{Z}^2$. Therefore the sphere at point x of radius n is the pattern defined on $B_1(x, n)$.

Both configurations x and y have the same patterns thus if a pattern appears only a finite number of times in x, it appears the same number of times in both configurations. As a consequence, Hanf lemma applies: x and y, having the same patterns, have the same theory. ■

This theorem allows us to get an equivalence between the completeness of T_τ and a property of the tileset τ:

Theorem 4.7. *A tileset τ can produce only one tiling up to local isomorphism if and only if $T_\tau^\infty = T_\tau \cup \{\forall x, E^n(N^m(x)) \neq x | m, n \in \mathbb{Z}\}$ is complete.*

Note that the additional axioms ensure that no model of T_τ is skewed, that is all models of T_τ are based on $\mathbb{Z}^2$ or disconnected copies of $\mathbb{Z}^2$.

There is no need indeed for these additional axioms if we can ensure that the only (up to local isomorphism) tiling by τ is actually strictly aperiodic (that is has no vector of periodicity)

Proof. Before going on the proof of Theorem 4.7, we first need an extra lemma on tilings:

Lemma 4.8. *If all the tilings produced by a tileset τ are locally isomorphic then every pattern that appears in a tiling appears infinitely many times in it.*

Proof. Consider a tiling x and suppose that there exists a pattern that appears only finitely many times. By compactness, we can extract a tiling that does not contain this pattern since we can have arbitrary large patterns that do not overlap with it. The extracted tiling that does not contain this pattern will thus not have the same patterns as x. ∎

$\Rightarrow$: We prove that any two models of T_τ^∞ are elementary equivalent. This is already true for models that are tilings (Lemma 4.6) but we still have to prove it for arbitrary models. Consider two models M and M' of T_τ^∞, they are made of disconnected copies of tilings; all patterns that appear in a tiling appear infinitely many times therefore all the spheres that appear in M or M' appear infinitely many times. Thus the hypothesis of Hanf locality lemma hold, so $M \equiv M'$. Therefore T_τ^∞ is complete.

$\Leftarrow$: If T_τ^∞ is complete, for any pattern M, the formula $\exists x, \varphi_M(x)$ is either valid in any model or false in any model, therefore any two tilings contain exactly the same patterns, thus τ can produce only one tiling up to local isomorphism. ∎

Corollary 4.9. *If a tileset τ can produce only one tiling up to local isomorphism then the appeareance of any pattern is a decidable problem.*

This is a corollary of Theorem 4.7 that we express here without any model theoretic language: τ can produce only one tiling up to local isomorphism thus T_τ^∞ is complete. Given a pattern M, one can enumerate the valid proofs in T_τ^∞ and stop when either a proof of $\exists x, \varphi_M(x)$ or of $\neg\exists x, \varphi_M(x)$ is found; and such a proof will be found since T_τ^∞ is complete.

4.2.1. *On compactness.* With all those results one could try to prove some results about tilings in an elegant and short way using model theoretic arguments. Take for example the fact that any tileset that produces only periodic tilings can produce only finitely many of them [1]. This can be reformulated as "if a tileset can produce tilings with arbitrarily large periods then it can produce one that is not periodic". It is easy to write a formula ϕ_n that expresses that there is a tiling with no period lower than n. If a tileset can produce tilings with arbitrarily large periods then it has a model verifying any finite set of such formulae, thus by compactness it has a model that verifies all these formulae, e.g., it has a model that has no period. However, we can not conclude that the tileset can produce a tiling with no period. Indeed this model we obtain by compactness will certainly consist of a copy of each periodic tiling : As we have tilings of arbitrary large period, there is no common period for all these tilings, so that our model indeed does not have a period.

We would like to be able to use the compactness theorem of the first order logic but within the domain of connected models. However as said earlier, many classical theorems of first order logic will not hold. See [7] for some possible solutions.

4.3. Applying the results to model theory

A finitely axiomatizable, complete and superstable theory has been exhibited with these methods of translating tilesets into first order theories. This has historically been done by

Makowsky [9] to prove that these three properties of theories are not incompatible and then explained in a more detailed way by Poizat [11].

The idea is quite simple: take τ an aperiodic tileset that produces only one tiling up to local isomorphism; for example the one used by Berger to prove the undecidability of the domino problem [2]. Transform it in a first order theory as explained in Section 3 to obtain a finitely axiomatized theory T_τ. Since Berger proved that his tileset can not produce any tiling with a vector of periodicity, Theorem 4.7 holds without any need to add more axioms to ensure that the models are infinite by Lemma 4.4; therefore T_τ is complete and finitely axiomatizable.

We can then prove that the theory is superstable. This definition has to do with how many types there are in the theory, or more simply, with how many tilings we can produce.

It has been proven that this tileset can produce $2^{\aleph_0}$ different tilings [4, 1], therefore $2^{\aleph_0}$ countable models; Those models are not isomorphic because there is only a countable number of shifts. Furthermore, there is no skewed models, so that all models of this theory are then easy to give : they consists of some copies of these $2^{\aleph_0}$ different tilings, that is we have to say for each tiling how many times it appears. This shows that the theory is not ω-stable, but superstable.

5. Conclusion

We have seen along this paper the tight links between tilings and logic, especially between tilings properties and model theoretical properties of their interpretation. Tilings have then provided interesting examples of theories [11] as well as a good framework in which to study properties of classes of formulaes[5]

Some links still remain unexplored and might lead to interesting results. As an exemple, the Cantor-Bendixson rank [8] introduced in [1] has been motivated by the study of a notion of rank for finitely generated structures of universal theories in [7].

References

[1] B. Durand A. Ballier, E. Jeandel. Structural aspects of tilings. *To be published in 25th International Symposium on Theoretical Aspects of Computer Science (STACS). Preliminary version in HAL.*, 2008.

[2] R. Berger. The undecidability of the domino problem. *Memoirs American Mathematical Society*, 66:1966, 1966.

[3] Julien Cervelle. *Complexité structurelle et algorithmique des pavages et des automates cellulaires.* PhD thesis, Université de Provence, 2002.

[4] Bruno Durand. Tilings and quasiperiodicity. *Theor. Comput. Sci.*, 221(1-2):61–75, 1999.

[5] Yuri Gurevich Ergon Börger, Erich Grädel. *The classical decision problem.* Springer-Verlag Telos, 1996.

[6] William Hanf. Model-theoretic methods in the study of elementary logic. *Symposium in the Theory of Models*, pages 132–145, 1965.

[7] Abderezak Ould Houcine. On finitely generated models of theories with at most countably many non isomorphic finitely generated models.

[8] Kazimierz Kuratowski. *Topology, Vol. I, 3rd edition.* NY: Academic Press, 1966.

[9] J.A. Makowsky. On some conjectures connected with complete sentences. *Fund. Math.*, 81:193–202, 1974.

[10] Francis Oger. Algebraic and model-theoretic properties of tilings. *Theoretical computer science*, 319:103–126, 2004.

[11] Bruno Poizat. Une theorie finiement axiomatisable et superstable. *Groupe d'etude des theories stables*, 3(1):1–9, 1980-1982.

[12] Hao Wang. Proving theorems by pattern recognition ii. *Bell system technical journal*, 40:1–41, 1961.

[13] Hao Wang. Dominoes and the aea case of the decision problem. *Mathematical theory of Automata*, pages 23–55, 1963.

[14] Hao Wang. Proving theorems by pattern recognition i. *The Journal of Symbolic Logic*, 32, 1967.

Journées Automates Cellulaires 2008 (Uzès), pp. 40-53

CLASSIFICATION OF DIRECTIONAL DYNAMICS FOR ADDITIVE CELLULAR AUTOMATA

A. DENNUNZIO [1], P. DI LENA [2], E. FORMENTI [3], AND L. MARGARA [2]

[1] Università degli Studi di Milano-Bicocca, Dipartimento di Informatica Sistemistica e Comunicazione, viale Sarca 336/14, 20126 Milano, Italy
E-mail address: dennunzio@disco.unimib.it

[2] Università degli Studi di Bologna, Dipartimento di Scienze dell'Informazione, via Mura Anteo Zamboni 7, 40127 Bologna, Italy
E-mail address: {dilena,margara}@cs.unibo.it

[3] Université de Nice-Sophia Antipolis, Laboratoire I3S, 2000 Route des Colles, 06903 Sophia Antipolis, France
E-mail address: enrico.formenti@unice.it

Abstract. We continue the study of cellular automata (CA) directional dynamics, *i.e.* the behavior of the joint action of CA and shift maps. This notion has been investigated for general CA in the case of expansive dynamics by Boyle and by Sablik for sensitivity and equicontinuity. In this paper we give a detailed classification for the class of additive CA providing non-trivial examples for some classes of Sablik's classification. Moreover, we extend the directional dynamics studies by considering also factor languages and attractors.

1. Introduction

Cellular automata (CA) are simple formal models for complex systems. They have been widely studied in a number of disciplines (Computer Science, Physics, Mathematics, Biology, Chemistry, *etc.*) with different purposes (simulation of natural phenomena, pseudo-random number generation, image processing, analysis of universal model of computations, quasi-crystals, *etc.*). For an extensive and up-to-date bibliography, for example, see [13, 8, 17, 21, 31, 11, 23].

The huge variety of distinct dynamical behaviors is one of the main features which determined the success of CA in applications. Paradoxically, the formal (decidable) classification of such behaviors is still a major open problem in CA theory. Indeed, many classifications have been introduced over the years but none of them is decidable [14, 9, 5, 19, 16, 12, 22].

This work has been supported by the Interlink/MIUR project "Cellular Automata: Topological Properties, Chaos and Associated Formal Languages", by the ANR Blanc "Projet Sycomore" and by the PRIN/MIUR project "Formal Languages and Automata: Mathematical and Applicative Aspects".

Inspired by [28, 4], M. Sablik proposed to refine Kůrka's equicontinuity classification along "directions" different from the standard time arrow [32]. The idea is to see how "robust" a given topological behavior is when changing the way by which time samples are taken into account. In other words, Sablik studies the space-time structure of CA evolutions by classifying the dynamics of $\sigma^k \circ F^h$, where σ is the shift map and F is the global rule of a CA ($k \in \mathbb{Z}, h \in \mathbb{N}^+$, see Section 2 for the definitions). Sablik's work is concerned particularly with directions of equicontinuity and (left/right) expansivity: he provides a directional dynamics classification of CA according to such properties. Despite his classification sheds new light about the complexity of CA behavior, most of his classes are still not well understood. Moreover, it is actually unknown whether his classification is (at least partially) decidable or not.

Additive CA (ACA) are the subclass of CA whose local rule is defined by an additive function. Despite their simplicity that makes it possible a detailed algebraic analysis, ACA exhibit many of the complex features of general CA. Several important properties of ACA have been studied during the last twenty years and in some cases exact characterizations have been obtained [15, 33, 27, 26, 7, 6].

In this paper we use ACA to further illustrate the work of Sablik and we extend the directional dynamics picture by further introducing attractors and factor languages directions. We provide a very detailed directional dynamics classification of ACA and we compare our classes with Sablik's ones. Moreover, we show that our classification is completely decidable.

The paper is organized as follows. Sections 2 to 4 are devoted to the basic background on the subject of CA and ACA. In Section 5, we consider factor languages directions, in particular we show that all ACA are regular. In Section 6 we consider attractor directions. In Section 7 we provide a directional dynamics classification of ACA and compare our classes with Sablik's ones. In Section 8, we draw some conclusions and provide arguments for the decidability of our classification.

For lack of space, proofs are put in the Appendix.

2. Cellular automata

A CA consists in an infinite set of finite automata distributed over a regular lattice $\mathcal{L}$. All finite automata are identical. Each automaton assumes a *state*, chosen from a finite set A, called the *set of states* or the *alphabet*. A *configuration* is a snapshot of all the states of the automata *i.e.* a function from $\mathcal{L}$ to A. In the present paper, we consider one dimensional CA in which $\mathcal{L} = \mathbb{Z}$. A *local rule* updates the state of each automaton on the basis of its current state and the ones of a fixed set of neighboring automata individuated by the neighborhood frame $N = \{m, m+1, \dots, a\}$, where $m, a \in \mathbb{Z}$, with $m \leq a$. The integers m, a and $r = \max\{|m|, |a|\}$ are called the *memory*, the *anticipation* and the *radius* of the CA, respectively. Formally, the local rule is a function $f : A^{a-m+1} \to A$. All automata in the lattice are updated synchronously. In other words, the local rule f induces a *global* rule $F : A^{\mathbb{Z}} \to A^{\mathbb{Z}}$ describing the evolution of the whole system from time t to $t+1$:

$$\forall c \in A^{\mathbb{Z}}, \forall i \in \mathbb{Z}, \quad F(c)_i = f(c_{i+m}, \dots, c_{i+a}) \ . \tag{2.1}$$

We say that a CA is *one-sided* if either $m \geq 0$ or $a \leq 0$. The *shift map* $\sigma : A^{\mathbb{Z}} \to A^{\mathbb{Z}}$, defined as $\forall c \in A^{\mathbb{Z}}$, $\forall i \in \mathbb{Z}$, $\sigma(c)_i = c_{i+1}$ is one of the simplest examples of CA (it is induced by the local rule $f : A^2 \to A$ defined as $\forall x_0, x_1 \in A,\ f(x_0, x_1) = x_1$ with memory $m = 0$ and anticipation $a = 1$).

In this work we restrict our attention to the class of additive CA, *i.e.*, CA based on an additive local rule defined over the ring $\mathbf{Z}_s = \{0, 1, \dots, s-1\}$. A function $f : \mathbf{Z}_s^{a-m+1} \to \mathbf{Z}_s$ is said to be additive if there exist $\lambda_m, \dots, \lambda_a \in \mathbf{Z}_s$ such that it can be expressed as:

$$\forall (x_m, \dots, x_a) \in \mathbf{Z}_s^{a-m+1}, \quad f(x_m, \dots, x_a) = \left[\sum_{j=m}^{a} \lambda_j x_j \right]_s$$

where $[x]_s$ is the integer x taken modulo s. A CA is *additive* if its local rule is additive. In this case, Equation (2.1) becomes

$$\forall c \in \mathbf{Z}_s^{\mathbb{Z}}, \forall i \in \mathbb{Z}, \quad F(c)_i = \left[\sum_{j=m}^{a} \lambda_j c_{i+j} \right]_s \quad .$$

A rule $f : A^{a-m+1} \to A$ is *permutive in the position i* if $\forall b_m, b_{m+1}, ..., b_{i-1}, b_{i+1}, .., b_a \in A, \forall b \in A, \exists! b_i \in A,\ f(b_m, ..., b_{i-1}, b_i, b_{i+1}, ..., b_a) = b$. The local rule of an ACA is permutive in the position i iff $gcd(s, \lambda_i) = 1$.

Finally, remark that if $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is additive, then for all $k \in \mathbb{Z}, h \in \mathbb{N}^+$ the automaton $(\mathbf{Z}_s^{\mathbb{Z}}, \sigma^k \circ F^h)$ is also additive.

3. Dynamical properties of topological dynamical systems and CA

A topological dynamical system is a pair (X, g) where X is a compact topological space and g is a continuous mapping from X to itself. When A is equipped with the discrete topology and $A^{\mathbb{Z}}$ with the induced product topology, for any CA F, the pair $(A^{\mathbb{Z}}, F)$ is a topological dynamical system. The study of the dynamical behavior of CA is interesting and captured the attention of researchers in the last decades. We now illustrate several properties which are widely recognized as fundamental in the characterization of the behavior of dynamical system.

Dynamical and set theoretical properties for topological dynamical systems. Let (X, g) be a topological dynamical system. It is *injective* (resp., *surjective*, *open*) iff g is injective (resp., surjective, open). It is *sensitive to the initial conditions* (or simply *sensitive*) if $\exists \varepsilon > 0$ such that $\forall x \in X, \forall \delta > 0, \exists y \in X$ such that $d(y, x) < \delta$ and $d(g^n(y), g^n(x)) > \varepsilon$ for some $n \in \mathbb{N}$. It is *positively expansive* if $\exists \varepsilon > 0$ such that $\forall x, y \in X, x \neq y, d(g^n(y), g^n(x)) \geq \varepsilon$ for some $n \in \mathbb{N}$. If X is a perfect set, any positively expansive dynamical system is also sensitive. When g is a homeomorphism it cannot be positively expansive. In this case the notion of *expansivity* can be considered. It is obtained by replacing $n \in \mathbb{N}$ with $n \in \mathbb{Z}$ in the definition of positive expansivity. Both sensitivity and expansivity are referred to as elements of unstability for the system. We now recall two notions which represent conditions of stability for topological dynamical systems. An element $x \in X$ is an *equicontinuity point* for g if $\forall \varepsilon > 0, \exists, \delta > 0$ such that $\forall y \in X, d(y, x) < \delta$ implies that $\forall n \in \mathbb{N},\ d(g^n(y), g^n(x)) < \varepsilon$. The dynamical systems (X, g) is said to be *equicontinuous* iff every point in X is an equicontinuity point. It is *almost equicontinuous* if the set of equicontinuity points is residual (*i.e.*, it can be obtained by an infinite intersection of dense open subsets).

The system (X, g) is *(topologically) transitive* if for any pair of non-empty open sets $U, V \subseteq X, \exists n \in \mathbb{N}$ such that $g^n(U) \cap V \neq \emptyset$. It is *(topologically) mixing* if for any pair of non-empty open sets $U, V \subseteq A^{\mathbb{Z}}, \exists n \in \mathbb{N}$ such that $\forall t \geq n, g^t(U) \cap V \neq \emptyset$. Trivially, a mixing dynamical system is transitive.

A morphism between two dynamical systems (X, g) and (Y, h) is a continuous map $\phi : X \to Y$ such that $h \circ \phi = \phi \circ g$. If ϕ is surjective, (Y, h) is a *factor* of (X, g). If ϕ is a homeomorphism, the two systems are said to be *(topologically) conjugated.* The conjugacy preserves most of the properties seen so far. In the sequel we recall some notions useful to understand the long term behavior of dynamical systems. For a given (X, g), a subset $V \subseteq X$ is said to be *invariant* if $g(V) \subseteq V$.The *omega limit* of a closed invariant subset $V \subseteq X$ is defined as

$$\omega(V) = \cap_{n>0}\overline{\cup_{m>n} g^m(V)}$$

The *limit set* of (X, g) is $\omega(X)$. A dynamical system is called *stable* if it reaches its limit set in a fine amount of time, i.e., if there exists some $n > 0$ such that $\forall m > n$, $g^m(X) = g^n(X)$. A set $Y \subseteq X$ is an attractor if there exists a nonempty open set V such that $F(\overline{V}) \subseteq V$ and $Y = \omega(V)$. In totally disconnected spaces, attractors are omega limit sets of clopen invariant sets. A set $Y \subseteq X$ is a *minimal* attractor if it is an attractor and no proper subset of Y is an attractor. A *quasi-attractor* is a countable intersection of attractors which is not an attractor.

Topology on CA configurations and related properties. In order to study the dynamical properties of CA, $A^{\mathbb{Z}}$ is usually equipped with the Cantor metric d defined as

$$\forall c, c' \in A^{\mathbb{Z}},\ d(c, c') = 2^{-n}, \text{ where } n = \min\left\{i \geq 0 \ : \ c_i \neq c'_i \text{ or } c_{-i} \neq c'_{-i}\right\} \ .$$

The topology induced by d coincides with the product topology defined above. In this case, $A^{\mathbb{Z}}$ is a Cantor space, *i.e.*, it is compact, perfect and totally disconnected.

For any configuration $c \in A^{\mathbb{Z}}$ and any pair $i, j \in \mathbb{Z}$, with $i \leq j$, denote by $c_{[i,j]}$ the word $c_i \cdots c_j \in A^{j-i+1}$, *i.e.*, the portion of the configuration $c \in A^{\mathbb{Z}}$ inside the interval $[i, j] = \{k \in \mathbb{Z} : i \leq k \leq j\}$. A *cylinder* of block $u \in A^k$ and position $i \in \mathbb{Z}$ is the set $C_i(u) = \{c \in A^{\mathbb{Z}} : c_{[i,i+k-1]} = u\}$. Cylinders are clopen (*i.e.* closed and open) sets *w.r.t.* the Cantor metric.

In the case of CA, it is possible to study other forms of expansivity. For any $n \in \mathbb{Z}$, let $c_{[n,\infty)}$ (resp., $c_{(-\infty,n]}$) denote the portion of a configuration c inside the infinite integer interval $[n, \infty)$ (resp., $(-\infty, n]$). A CA $(A^{\mathbb{Z}}, F)$ is *right* (resp., *left*) *expansive* if there exists a constant $\varepsilon > 0$ such that for any pair of configurations $c, c' \in A^{\mathbb{Z}}$ with $c_{[0,\infty)} \neq c'_{[0,\infty)}$ (resp., $c_{(-\infty,0]} \neq c'_{(-\infty,0]}$) we have $d(F^n(c), F^n(c')) \geq \varepsilon$ for some $n \in \mathbb{N}$. Remark that a CA is positively expansive iff it is both left and right expansive. A simple class of left (resp., right) expansive CA is the one of automata whose local rule is permutive in its leftmost (resp., rightmost) position.

Subshifts and column subshifts. A *subshift* on the alphabet A is a pair (S, σ) where S is a closed σ-invariant subset of the *full shift* $A^{\mathbb{N}}$ (or $A^{\mathbb{Z}}$). From now on, for the sake of simplicity, when it is clear from the context, we identify a subshift (S, σ) with the set S. For $w = w_1 \cdots w_n \in A^n$ and $y \in A^{\mathbb{N}}$, $w \prec y$ means that w is a proper factor of $y \in A^{\mathbb{N}}$, *i.e.*, there exists $i \in \mathbb{N}$ such that $y_{[i,i+n-1]} = w$. Let $\mathcal{F} \subseteq A^*$ and $S_{\mathcal{F}} = \left\{y \in A^{\mathbb{N}} : \forall w \prec y,\, w \notin \mathcal{F}\right\}$. $S_{\mathcal{F}}$ is a subshift, and $\mathcal{F}$ is its set of *forbidden patterns.* A subshift S is said to be a *subshift of finite type* (SFT) if $S = S_{\mathcal{F}}$ for some finite set $\mathcal{F}$. The language of a subshift S is $L_S = \left\{w \in A^* : \exists y \in A^{\mathbb{N}}, w \prec y\right\}$. A subshift is *sofic* if it is a factor of some SFT. We refer to [24] for more on subshifts. Let S_1 and S_2 be two subshifts. A function $\varphi : S_1 \to S_2$ is said to be a *block map* if it is continuous and σ-commuting, i.e., $\varphi \circ \sigma = \sigma \circ \varphi$. In particular, CA are block maps from the sushift $A^{\mathbb{Z}}$ to itself. The *column subshift* of width $k > 0$ of a given CA $(A^{\mathbb{Z}}, F)$, is the subshift $(\Sigma_k(F), \sigma)$ on the $B = A^k$ where

$$\Sigma_k(F) = \left\{ y \in B^{\mathbb{N}} : \exists c \in A^{\mathbb{Z}}, \forall i \in \mathbb{N}, y_i = F^i(c)_{[1,k]} \right\} .$$

Remark that, for a given CA F, the set $\Sigma_k(F)$ does not change when replacing the interval $[1, k]$ involved in the previous definition with any other interval of width k . A language $L \subseteq A^*$ is *bounded periodic* if there exist two integers $l \geq 0$ and $n > 0$ such that for every $u \in L$ and $i \geq l$ we have $u_i = u_{i+n}$. A CA is said to be *bounded periodic* (resp., *regular*) if for any $k > 0$ the the language of the column subshift $(\Sigma_k(F), \sigma)$ is bounded periodic (resp., regular).

Directional dynamics of CA. The directional dynamics of CA concerns the study of the joint action of CA with the shift map. More precisely, for a given CA F and for any rational $k/h \in \mathbb{Q}$, the focus is the dynamical behavior of the CA $\sigma^k F^h$. A CA F is said to be equicontinuous (resp., almost equicontinuous, resp. left expansive, resp., right expansive, resp., positively expansive, resp., expansive) along the direction k/h, $k \in \mathbb{Z}$, $h \in \mathbb{N}^+$, if the CA $\sigma^k F^h$ is equicontinuous (resp., almost equicontinuous, resp. left expansive, resp., right expansive, resp., positively expansive, resp., expansive). Note that all the above properties are preserved along directions, i.e., if $\sigma^k F^h$ has property $\mathcal{P}$ then $\forall n > 0, (\sigma^k F^h)^n$ has property $\mathcal{P}$.

3.1. Classifications of CA

This section reviews three important classifications of CA based on the complexity of their column subshift languages, the degree of stability/unstability of their behavior, and the existence of attractors, respectively. All these classifications have been defined and compared in [19].

Theorem 3.1. [19] *Every CA $(A^{\mathbb{Z}}, F)$ falls exactly in one of the following classes:*

L1 $(A^{\mathbb{Z}}, F)$ *is bounded periodic.*
L2 $(A^{\mathbb{Z}}, F)$ *is regular not bounded periodic.*
L3 $(A^{\mathbb{Z}}, F)$ *is not regular.*

Theorem 3.2. [19] *Every CA $(A^{\mathbb{Z}}, F)$ falls exactly in one of the following classes:*

E1 $(A^{\mathbb{Z}}, F)$ *is equicontinuous;*
E2 $(A^{\mathbb{Z}}, F)$ *is almost equicontinuous but not equicontinuous;*
E3 $(A^{\mathbb{Z}}, F)$ *is sensitive but not positively expansive;*
E4 $(A^{\mathbb{Z}}, F)$ *is positively expansive.*

Factor languages of equicontinuous and positively expansive CA have been studied in deep. Here we just recall some results that will be useful later.

Theorem 3.3. [19] **L1** = **E1**.

Theorem 3.4. [19, 29] *Let $(A^{\mathbb{Z}}, F)$ be a positively expansive CA with memory and anticipation $m < 0 < a$. Then, it is conjugated to $(\Sigma_{a-m+1}(F), \sigma)$ which is a mixing SFT.*

In particular, Nasu proved that $(\Sigma_{a-m+1}(F), \sigma)$ is conjugated to a full shift [29].

Theorem 3.5. [3] *Let $(A^{\mathbb{N}}, F)$ be a positively expansive CA with anticipation $a > 0$. Then, it is conjugated to $(\Sigma_a(F), \sigma)$ which is a mixing SFT.*

Theorem 3.6. [19] *Every CA $(A^{\mathbb{Z}}, F)$ falls exactly in one of the following classes.*

A1 *There exist two disjoint attractors.*
A2 *There exists a unique minimal quasi-attractor.*
A3 *There exists a unique minimal attractor different from $\omega(A^{\mathbb{Z}})$.*
A4 *There exists a unique attractor $\omega(A^{\mathbb{Z}}) \neq A^{\mathbb{Z}}$.*
A5 *There exists a unique attractor $\omega(A^{\mathbb{Z}}) = A^{\mathbb{Z}}$.*

We report some results concerning attractors of CA. They will be useful in the sequel.

Theorem 3.7. [20] *An equicontinuous CA has either a pair of disjoint attractors or a unique attractor which is a singleton.*

If an equicontinuous CA is surjective then it must have two disjoint attractors. CA with a unique attractor which is a singleton are called *nilpotent*.

Theorem 3.8. [19] *A transitive CA has a unique attractor.*

Theorem 3.9. [1] *Let $(A^{\mathbb{Z}}, F)$ be a surjective CA with memory m and anticipation a. If either $m > 0$ or $a < 0$, then F is mixing.*

Since transitive CA are surjective and mixing CA are transitive (see [20], for example), from Theorem 3.7 and Theorem 3.9 it follows that surjective CA with either memory $m > 0$ or anticipation $a < 0$ have a unique attractor.

A recent classification concerns the directional dynamics of a CA F. In order to illustrate it, we introduce the following notation.

Definition 3.10. The *equicontinuous, almost equicontinuous, expansive* and *left-or-right expansive* direction sets of a CA $(A^{\mathbb{Z}}, F)$ are defined as follows

- $\mathfrak{E}_F = \{k/h \mid k \in \mathbb{Z}, h \in \mathbb{N}^+ : \sigma^k F^h$ is equicontinuous$\}$.
- $\mathfrak{A}_F = \{k/h \mid k \in \mathbb{Z}, h \in \mathbb{N}^+ : \sigma^k F^h$ is almost equicontinuous$\}$.
- $\mathfrak{X}_F^- = \{k/h \mid k \in \mathbb{Z}, h \in \mathbb{N}^+ : \sigma^k F^h$ is left expansive$\}$.
- $\mathfrak{X}_F^+ = \{k/h \mid k \in \mathbb{Z}, h \in \mathbb{N}^+ : \sigma^k F^h$ is right expansive$\}$.
- $\mathfrak{X}_F = \{k/h \mid k \in \mathbb{Z}, h \in \mathbb{N}^+ : \sigma^k F^h$ is expansive$\}$.

Note that the sets $\mathfrak{E}_F, \mathfrak{A}_F, \mathfrak{X}_F^-$ and $\mathfrak{X}_F^+$ are convex (in $\mathbb{Q}$ or in $\mathbb{R}$). Moreover, note that the set of positively expansive directions is $\mathfrak{X}_F^+ \cap \mathfrak{X}_F^-$.

Theorem 3.11. [32] *Let $(A^{\mathbb{Z}}, F)$ be a CA with memory m and anticipation a.*

- *If $|\mathfrak{E}_F| > 1$ then $\mathfrak{E}_F = \mathbb{Q}$ and $(A^{\mathbb{Z}}, F)$ is nilpotent.*
- *If $\mathfrak{E}_F \neq \emptyset$ and $\mathfrak{E}_F \neq \mathbb{Q}$ then $\exists! \alpha \in [-a, -m], \mathfrak{E}_F = \{\alpha\}$ and $\mathfrak{X}_F^- = (-\infty, \alpha)$, $\mathfrak{X}_F^+ = (\alpha, \infty)$. In particular, $(A^{\mathbb{Z}}, F)$ is injective.*

Theorem 3.12. [32] *Every $(A^{\mathbb{Z}}, F)$ CA with memory m and anticipation a falls exactly in one of the following classes:*

C1. *$\mathfrak{E}_F = \mathfrak{A}_F = \mathbb{Q}$ and $\mathfrak{X}_F^- = \mathfrak{X}_F^+ = \emptyset$. This happens iff $(A^{\mathbb{Z}}, F)$ is nilpotent.*
C2. *There exists $\alpha \in [-a, -m]$, $\mathfrak{E}_F = \mathfrak{A}_F = \{\alpha\}$. Moreover, if $(A^{\mathbb{Z}}, F)$ is surjective, $\mathfrak{X}_F^- = (-\infty, \alpha)$ and $\mathfrak{X}_F^+ = (\alpha, \infty)$.*
C3. *There exists $\alpha \in [-a, -m]$, $\mathfrak{E}_F = \emptyset, \mathfrak{A}_F = \{\alpha\}$.*

C4. *There exist $\alpha_1 < \alpha_2$ such that $(\alpha_1, \alpha_2) \subseteq \mathfrak{A}_F \subseteq [\alpha_1, \alpha_2] \subseteq [-a, -m]$ and $\mathfrak{C}_F = \mathfrak{X}_F^- = \mathfrak{X}_F^+ = \emptyset$.*
C5. *$\mathfrak{X}_F^- \cap \mathfrak{X}_F^+ \neq \emptyset$. This implies $\mathfrak{C}_F = \mathfrak{A}_F = \emptyset$.*
C6. *$\mathfrak{C}_F = \mathfrak{A}_F = \emptyset$ and $\mathfrak{X}_F^- \cap \mathfrak{X}_F^+ = \emptyset$.*

3.2. Main properties of ACA

The dynamical behavior of ACA has been extensively studied. We briefly report the main results which characterize the most important dynamical and set theoretical properties for ACA.

Theorem 3.13. [15, 25, 27, 7, 6] *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an ACA with local rule $f(x_m, \ldots, x_a) = \left[\sum_{j=-m}^{a} \lambda_j x_j\right]_s$ and with $s = p_1^{n_1} \cdot p_2^{n_2} \cdot \ldots \cdot p_l^{n_l}$ where $p_1, .., p_l$ are primes. Then,*

- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is surjective iff $\gcd(s, \lambda_{-m}, ..., \lambda_a) = 1$*
- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is injective iff $\forall p_i, \exists! \lambda_j, p_i \nmid \lambda_j$*
- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is equicontinuous iff $\forall p_i, p_i \mid \gcd(\lambda_{-m}, ..., \lambda_{-1}, \lambda_1, ..., \lambda_a)$*
- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is sensitive iff $\exists p_i, p_i \nmid \gcd(\lambda_{-m}, ..., \lambda_{-1}, \lambda_1, ..., \lambda_a)$*
- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is transitive iff it is mixing iff $\gcd(s, \lambda_{-m}, ...\lambda_{-1}, \lambda_1, ..., \lambda_a) = 1$*
- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is pos. expansive iff $\gcd(s, \lambda_{-m}, ..., \lambda_{-1}) = \gcd(s, \lambda_1, ..., \lambda_a) = 1$*
- *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is expansive iff $\gcd(s, \lambda_{-m}, ..., \lambda_{-1}, \lambda_1, ..., \lambda_a) = 1$*

Remark that, as immediate consequence of Theorem 3.13, **E2** $= \emptyset$ for ACA. Moreover, all the characterizations are given in terms of coefficients of the local rule and hence they are decidable.

In the sequel, we are going to recall two fundamental tools. The former states that any ACA has a canonical decomposition into simple basic ACA. The latter tells us that in order to study ACA one can focus on the surjective ones. This is possible since ACA are stable and the class of possible dynamics on their limit sets is equivalent to the class of dynamics of surjective ACA.

Theorem 3.14. [10] *Consider an ACA $(\mathbf{Z}_{pq}^{\mathbb{Z}}, F)$ with $\gcd(p, q) = 1$. Then $(\mathbf{Z}_{pq}^{\mathbb{Z}}, F)$ is conjugated to the ACA $(\mathbf{Z}_p^{\mathbb{Z}} \times \mathbf{Z}_q^{\mathbb{Z}}, [F]_p \times [F]_q)$.*

On the basis of this theorem, if $s = p_1^{n_1} \cdots p_l^{n_l}$ is the prime factor decomposition of s, an ACA on $\mathbf{Z}_s$ is conjugated to the product of ACA on $\mathbf{Z}_{p_i^{n_i}}$. So all the properties which are preserved under product and under topological conjugacy are lifted from ACA on $\mathbf{Z}_{p^k}$ to $\mathbf{Z}_s$.

Theorem 3.15. [26] *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an ACA. Then $\forall h \geq \lfloor \log_2 s \rfloor$, $F^h(\mathbf{Z}_s^{\mathbb{Z}}) = F^{\lfloor \log_2 s \rfloor}(\mathbf{Z}_s^{\mathbb{Z}}) = \omega(\mathbf{Z}_s^{\mathbb{Z}})$. Moreover, $(F^h(\mathbf{Z}_s^{\mathbb{Z}}), F)$ is conjugated to some surjective ACA $(\mathbf{Z}_{s*}^{\mathbb{Z}}, F*)$.*

Remark that the conjugacy map involved in the proof of Theorem 3.15 preserves factor languages complexities, i.e., for $k > 0$, the column factor of width k of $(F^h(\mathbf{Z}_s^{\mathbb{Z}}), F)$ is a SFT if and only if the column factor of width k of $(\mathbf{Z}_{s*}^{\mathbb{Z}}, F*)$ is SFT. This property will be useful in the sequel.

4. Surjective ACA

Thanks to Theorem 3.14 and Theorem 3.15, most of the properties of general ACA can be deduced from *undecomposable ACA*, namely surjective ACA over $\mathbf{Z}_{p^k}$ for some prime number p. In this section we restrict our attention to the class of surjective ACA and, in particular, we classify the possible dynamics of undecomposable surjective ACA. The results contained in this section will be useful later to understand the directional dynamics of general ACA.

One useful property of undecomposable ACA is that there always exist powers of the *undecomposable maps* which are permutive in both their leftmost and rightmost positions.

Lemma 4.1. *Let $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ be a surjective ACA with p prime whose local rule has memory m and anticipation a. Then there exists $i \in [m, a]$ such that* $\gcd(\lambda_i, p) = 1$.

Lemma 4.2. [10] *Let $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ be a surjective ACA with p prime. Set*

$$L = \min\{j : \gcd(\lambda_j, p) = 1\} \quad \text{and} \quad R = \max\{j : \gcd(\lambda_j, p) = 1\}.$$

Then there exists $h \geq 1$ such that the local rule f^h associated to F^h has the form

$$f^h(x_{hm}, ..., x_{ha}) = \left[\Sigma_{i=hL}^{hR} \mu_i x_i\right]_{p^k} \quad \text{with } \gcd(\mu_{hL}, p) = \gcd(\mu_{hR}, p) = 1.$$

Recall that the condition $\gcd(\mu_{hL}, p) = \gcd(\mu_{hR}, p) = 1$ implies permutivity in hL and hR. The following proposition characterizes the possible dynamics of undecomposable CA.

Proposition 4.3. *Consider a surjective ACA $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ with p prime. Then, exactly one of the following cases occurs:*

1. $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ *is equicontinuous.*
2. $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ *is positively expansive.*
3. $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ *is either left or right expansive.*

Remark 4.4. By the same property used in the previous proof, one can show that right/left expansive ACA on $\mathbf{Z}_{p_i^{n_i}}$ are mixing.

The following theorem classifies the directional dynamics of undecomposable surjective ACA: any undecomposable ACA either contains exactly one equicontinuous direction (and it is injective) or contains a positively expansive direction (and it is not injective).

Theorem 4.5. *Let $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ be a surjective ACA with p prime. Then exactly one of the following cases can occur*

1. $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ *is injective. Then,*
$$\mathfrak{X}_F^- \cap \mathfrak{X}_F^+ = \emptyset, |\mathfrak{E}_F| = 1 \text{ and } \mathfrak{X}_F = \mathfrak{X}_F^- \cup \mathfrak{X}_F^+ = \mathbb{Q} \setminus \mathfrak{E}_F.$$
2. $(\mathbf{Z}_{p^k}^{\mathbb{Z}}, F)$ *is not injective. Then,*
$$\mathfrak{X}_F^- \cap \mathfrak{X}_F^+ \neq \emptyset \text{ and } \mathfrak{E}_F = \mathfrak{X}_F = \emptyset.$$

Remark 4.6. Since the openness property is preserved in every direction and it is preserved also under product, by Theorem 3.14 and Theorem 4.5 it follows that any surjective ACA is open. For proofs of this property in a more general setting see, for example, [33] and [18].

5. Directional dynamics of ACA according to regularity

In this section we show that all ACA are regular. This fact implies that the dynamics of ACA is regular in all rational directions.

Theorem 5.1. [30] *A subshift $\Sigma \subseteq A^{\mathbb{N}}$ is a SFT if and only if $\sigma : \Sigma \to \Sigma$ is open.*

Lemma 5.2. *Let $\Sigma \subseteq A^{\mathbb{N}}$ be a subshift. Then the following conditions are equivalent:*

1. *(Σ, σ) is open*
2. *$\forall n > 0, (\Sigma, \sigma^n)$ is open.*
3. *$\exists n > 1$ such that (Σ, σ^n) is open.*

A proof of Lemma 5.2 in a more general setting can be found in [2].

Lemma 5.3. *Let $(\mathbf{Z}_{p^n}^{\mathbb{Z}}, F)$ be a right (left) expansive ACA with p prime. Then for all sufficiently large k, $(\Sigma_k(F), \sigma)$ is a SFT.*

Note that the condition that for all sufficiently large $k > 0, \Sigma_k(F)$ is a SFT is sufficient to conclude that $(\mathbf{Z}_{p^n}^{\mathbb{Z}}, F)$ is regular.

Theorem 5.4. *Any ACA is regular.*

Actually, since the conjugacy of Theorem 3.15, preserves factor languages, we can obtain the following more strong property.

Corollary 5.5. *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an ACA. Then for all sufficiently large $k > 0, \Sigma_k(F)$ is a SFT.*

Question 1. Is there any ACA having a strictly sofic column factor $\Sigma_k(F)$?

6. Directional dynamics of ACA according to attractors

In this section we study the class of attractors of ACA according to rational directions. In [26], Manzini and Margara show that any ACA can have either a unique attractor or a pair of disjoint attractors. Here we show some properties of *disjoint attractor* directions of ACA. We will need the two following results.

Lemma 6.1. *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be a surjective ACA and let $s = p_1^{n_1} \cdot p_2^{n_2} \cdot ... \cdot p_l^{n_l}$ be the prime factor decomposition of s. Then the following conditions are equivalent:*

1. *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ has two disjoint attractors,*
2. *$(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is not mixing,*
3. *$(\mathbf{Z}_{p_i^{n_i}}^{\mathbb{Z}}, [F]_{p_i^{n_i}})$ is equicontinuous for some $p_i^{n_i}$.*

We can easily characterize the class of attractors of ACA from the class of attractors of surjective undecomposable ACA.

Theorem 6.2. [26] *Any ACA has either a unique attractor or a pair of disjoint attractors.*

We can now study the set of *disjoint attractor directions* of ACA.

Definition 6.3. Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an ACA. The *disjoint attractors* direction set of $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is

$$\mathfrak{D}_F = \{k/h \mid k \in \mathbb{Z}, h \in \mathbb{N}^+ : \sigma^k F^h \text{ has two disjoint attractors}\}.$$

The following proposition shows some properties of the set $\mathfrak{D}_F$. In particular, we have that $\mathfrak{D}_F$ is finite and that between two disjoint attractors directions $\alpha_1, \alpha_2 \in \mathfrak{D}_F$ there cannot exist left/right expansive directions.

Proposition 6.4. *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an ACA with memory m and anticipation a. Then the following conditions hold.*

1. *If $|\mathfrak{E}_F| > 1$ then $\mathfrak{D}_F = \emptyset$.*
2. *If $\mathfrak{E}_F = \{\alpha\}$ then $\mathfrak{D}_F = \{\alpha\}$.*
3. *If $|\mathfrak{D}_F| > 1$ then $\mathfrak{E}_F = \emptyset$.*
4. *$\mathfrak{D}_F \subset [-a, -m]$ is finite.*
5. *If $\mathfrak{D}_F = \{\alpha_1, ..., \alpha_n\}$ then $\forall \alpha_i \leq \alpha_j, [\alpha_i, \alpha_j] \not\subset \mathfrak{X}_F^- \cup \mathfrak{X}_F^+$.*

To conclude we enumerate some classes of ACA for which $\mathfrak{D}_F$ is easy to characterize.

Corollary 6.5. *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an ACA.*

- *If $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is nilpotent then $\mathfrak{D}_F = \emptyset$.*
- *If $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is equicontinuous and not nilpotent then $\mathfrak{D}_F = \{0\}$.*
- *If $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is positively expansive then $\mathfrak{D}_F = \emptyset$.*
- *If $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is expansive then $\mathfrak{D}_F \neq \emptyset$*

In the case of ACA, the presence of a direction with two disjoint attractors is tightly linked to the presence of some form of equicontinuity. Indeed, such an ACA is either equicontinuous (not nilpotent) or it is the product of an ACA having an equicontinuous direction with some other ACA (see Lemma 6.1). It is not known if the same holds for general CA.

7. Directional dynamics of ACA

In this section we classify the directional dynamics of ACA according to equicontinuous, left/right expansive, expansive and disjoint attractor directions. We do not consider explicitly factor languages directions since, by Theorem 5.4, for ACA all language directions are regular, and, by Theorem 3.3, directions which have bounded periodic languages coincide exactly with equicontinuous directions. To have a more clear picture we introduce explicitly the class of strictly sensitive nonexpansive directions.

Definition 7.1. The *strictly sensitive* direction sets of the ACA $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ is defined by $\mathfrak{S}_F = \mathbb{Q} \setminus (\mathfrak{E}_F \cup \mathfrak{X}_F^- \cup \mathfrak{X}_F^+ \cup \mathfrak{X}_F)$.

We consider separately the directional dynamics of non surjective, strictly surjective and injective ACA. Note that, since there are no almost equicontinuous ACA, classes **C3** and **C4** of Theorem 3.12 are empty for ACA. By Theorem 4.5, it follows that surjective ACA always have left and right expansive directions. In particular, it is not difficult to see that for any surjective ACA of memory m and anticipation a it happens that $(-\infty, -a) \subseteq \mathfrak{X}_F^-$ and $(-m, \infty) \subseteq \mathfrak{X}_F^+$. This implies that surjective ACA can only belong to classes **C2**, **C5**, **C6**. In particular, injective ACA are contained in class **C2** $\cup$ **C6** and strictly surjective ACA are contained in **C5** $\cup$ **C6**. Obviously, in the strictly surjective case there are not expansive directions which arise uniquely in the injective case. For injective ACA it happens also that $\mathfrak{D}_F \neq \emptyset$ and that expansive directions are always the complement in $\mathbb{Q}$ of $\mathfrak{D}_F$.

Theorem 7.2. *Let $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ be an injective ACA with memory m and anticipation a. Then $\mathfrak{X}_F = \mathbb{Q} \setminus \mathfrak{D}_F$. Moreover, exactly one of the following cases can occur:*

1. $\mathfrak{E}_F \neq \emptyset$. *Then* $\mathfrak{D}_F = \mathfrak{E}_F = \{\alpha\} \subset [-a,-m]$, $\mathfrak{X}^+{}_F = (\alpha,\infty)$, $\mathfrak{X}^-{}_F = (-\infty,\alpha)$.
2. $\mathfrak{E}_F = \emptyset$. *Then* $\mathfrak{D}_F = \{\alpha_1,..,\alpha_n\} \subset [-a,-m]$, *with* $\alpha_1 < .. < \alpha_n, n > 1$ *and* $\mathfrak{X}^-{}_F = (-\infty,\alpha_1), \mathfrak{X}^+{}_F = (\alpha_n,\infty)$.

Strictly surjective ACA trivially cannot contain equicontinuous directions but they can have disjoint attractors directions.

Theorem 7.3. *Let* $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *be a surjective but non injective ACA with memory* m *and anticipation* a. *Then* $\mathfrak{E}_F = \emptyset$. *Moreover, exactly one of the following cases occurs.*

1. $\mathfrak{D}_F = \emptyset$ *and* $\mathfrak{X}^-{}_F \cap \mathfrak{X}^+{}_F = \emptyset$. *Then* $\exists \alpha_1,\alpha_2 \in [-a,-m], \alpha_1 < \alpha_2, \mathfrak{X}^-{}_F = (-\infty,\alpha_1)$, $\mathfrak{X}^+{}_F = (\alpha_2,\infty)$, $\mathfrak{S}_F = [\alpha_1,\alpha_2]$.
2. $\mathfrak{D}_F = \emptyset$ *and* $\mathfrak{X}^-{}_F \cap \mathfrak{X}^+{}_F \neq \emptyset$. *Then* $\exists \alpha_1,\alpha_2 \in [-a,-m]$, $\alpha_2 \le \alpha_1$, $\mathfrak{X}^-{}_F = (-\infty,\alpha_1)$, $\mathfrak{X}^+{}_F = (\alpha_2,\infty)$, $\mathfrak{S}_F = \emptyset$.
3. $\mathfrak{D}_F \neq \emptyset$. *Then* $\exists -a \le \alpha_1 \le \beta_1 \le .. \le \beta_n \le \alpha_2 \le -m, \mathfrak{D}_F = \{\beta_1,..,\beta_n\}$, $\mathfrak{X}^-{}_F = (-\infty,\alpha_1), \mathfrak{X}^+{}_F = (\alpha_2,\infty), \mathfrak{S}_F = [\alpha_1,\alpha_2]$.

For any non surjective CA trivially $\mathfrak{X}^-{}_F = \mathfrak{X}^+{}_F = \mathfrak{X}_F = \emptyset$.

Theorem 7.4. *Let* $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *be a non surjective ACA. Then exactly one of the following cases can occur.*

1. $\mathfrak{E}_F = \mathbb{Q}$ *and* $\mathfrak{D}_F = \mathfrak{S}_F = \emptyset$.
2. $\mathfrak{E}_F = \mathfrak{D}_F = \{\alpha\} \subseteq [-a,-m]$ *and* $\mathfrak{S}_F = \mathbb{Q} \setminus \{\alpha\}$.
3. $\mathfrak{S}_F = \mathbb{Q}, \mathfrak{E}_F = \emptyset$ *(with either* $\mathfrak{D}_F = \emptyset$ *or* $\mathfrak{D}_F \neq \emptyset$*)*.

As requested by one of the referee, in the next theorem we summarise all our results and we express them in terms of the coefficients of the local rule. We beg the reader pardon for its unreadable form.

Theorem 7.5. *Let* $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *be an ACA with local rule* $f(x_m,\ldots,x_a) = \left[\sum_{j=m}^{a} \lambda_j x_j\right]_s$ *and with* $s = p_1^{n_1} \cdot p_2^{n_2} \cdot \ldots \cdot p_l^{n_l}$ *where* $p_1,..,p_l$ *are primes. Then,*

1.1 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 1. *of Theorem 7.2 iff*
$$\exists! \lambda_j, \forall p_i, p_i \nmid \lambda_j$$
1.2 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 2. *of Theorem 7.2 iff*
$$\forall p_i, \exists! \lambda_j, p_i \nmid \lambda_j \text{ and } \nexists! \lambda_j, \forall p_i, p_i \nmid \lambda_j$$
2.1 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 1. *of Theorem 7.3 iff*
$$\forall p_i, \exists \lambda_{j'} \neq \lambda_{j''}, p_i \nmid \lambda_{j'}, p_i \nmid \lambda_{j''} \text{ and } \nexists k \in [m,a], \forall p_i, \exists \lambda_{j'} < k \le \lambda_{j''}, p_i \nmid \lambda_{j'}, p_i \nmid \lambda_{j''}$$
2.2 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 2. *of Theorem 7.3 iff*
$$\exists k \in [m,a], \forall p_i, \exists \lambda_{j'} < k \le \lambda_{j''}, p_i \nmid \lambda_{j'}, p_i \nmid \lambda_{j''}$$
2.3 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 3. *of Theorem 7.3 iff*
$$\forall p_i, \exists \lambda_j, p_i \nmid \lambda_j \text{ and } \exists p_i, \exists! \lambda_j, p_i \nmid \lambda_j \text{ and } \exists p_{i'}, \exists \lambda_{j'} \neq \lambda_{j''}, p_{i'} \nmid \lambda_{j'}, p_{i'} \nmid \lambda_{j''}$$
3.1 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 1. *of Theorem 7.4 iff it is nilpotent iff*
$$\forall p_i, \forall \lambda_j, p_i \mid \lambda_j$$
3.2 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 2. *of Theorem 7.4 iff*
$$\gcd(s,\lambda_m,...,\lambda_a) \neq 1 \text{ and } \exists p_i, \exists \lambda_j, p_i \nmid \lambda_j \text{ and}$$
$$\exists k \in [m,a], \forall p_i, p_i \mid \gcd(\lambda_m,...,\lambda_{k-1},\lambda_{k+1},...,\lambda_a)$$
3.3 $(\mathbf{Z}_s^{\mathbb{Z}}, F)$ *is in class* 3. *of Theorem 7.4 iff*

$$\gcd(s, \lambda_m, ..., \lambda_a) \neq 1 \ \textit{and} \ \exists p_i, \exists \lambda_j, p_i \nmid \lambda_j \ \textit{and}$$
$$\nexists k \in [m, a], \forall p_i, p_i \mid \gcd(\lambda_m, ..., \lambda_{k-1}, \lambda_{k+1}, ..., \lambda_a)$$

8. Conclusions

In this paper we have completely characterized the directional dynamics of ACA, not only *w.r.t.* equicontinuity or expansivity (as in the Sablik's approach) but also *w.r.t.* attractors and factor languages. Figures 1 to 3 summarize all the possibles scenarios.

Looking at the pictures, one immediately sees that the algebraic nature of ACA has greatly reduced the number and complexity of the possible dynamics. For example, we have proved that the factor languages of any ACA are regular along any direction. Of course, this is not true for the general case but it would be very interesting to investigate which is the largest class of CA with such a property.

The directional classification proposed by Sablik [32] sheds some light on how information propagates space-time diagrams of CA. For instance, there is no exchange of information between zones delimited by two directions of equicontinuity (almost equicontinuity) and the rest of phase space. In this paper, we showed that this is also the case for CA having directions with two disjoint attractors (see Figure 2 right or Figure 3). Remark that in the case of ACA, directions with two disjoint attractors are always tightly linked to the presence of equicontinuity (see Lemma 6.1). We wonder if this happens also in the general case where the situation is much more complicated, since one must take into account also almost equicontinuity and other types of attractors.

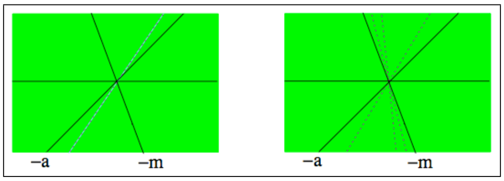


Figure 1: Directional dynamics for injective ACA. Green area depicts expansive directions. Lightblue line is a direction of equicontinuity. Magenta dotted lines are directions presenting two disjoint attractors.

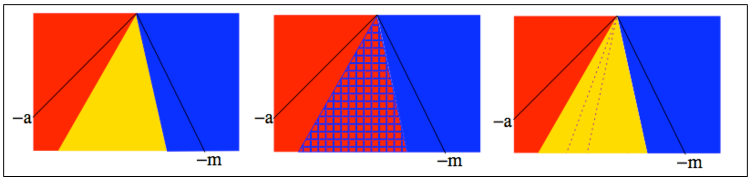


Figure 2: Directional dynamics for surjective ACA. Red (resp., blue) area depicts left (resp., right) expansive directions. Gold area indicates directions presenting sensitivity. Red/blue checkered area gives the positively expansive directions. Magenta dotted lines are directions presenting two disjoint attractors.

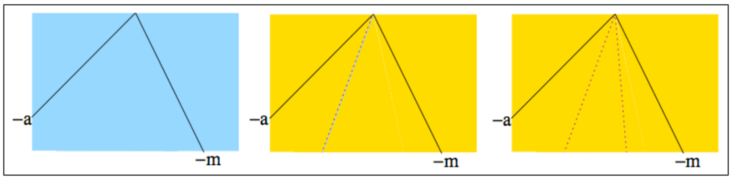


Figure 3: Directional dynamics for non-surjective ACA. Lightblue (resp., gold) area indicates equicontinuity (resp. sensitivity) directions. Magenta dotted lines are directions presenting two disjoint attractors.

To conclude we remark that, from Theorem 7.5, it follows that our classification is completely decidable.

References

[1] L. Acerbi, A. Dennunzio, and E. Formenti. Shifting and lifting of cellular automata. In *Third Conference on Computability in Europe, CiE 2007, Siena, Italy, June 18-23, 2007*, volume 4497 of *Lecture Notes in Computer Science*, pages 1–10. Springer Verlag, 2007.

[2] L. Acerbi, A. Dennunzio, and E. Formenti. Conservation of some dynamical properties for operations on cellular automata. Preprint, 2008.

[3] F. Blanchard and A. Maass. Dynamical properties of expansive one-sided cellular automata. *Israel Journal of Mathematics*, 99:149–174, 1997.

[4] M. Boyle and D. Lind. Expansive subdynamics. *Transactions of the American Mathematical Society*, 349:55–102, 1997.

[5] G. Braga, G. Cattaneo, P. Flocchini, and C. Quaranta Vogliotti. Pattern growth in elementary cellular automata. *Theoretical Computer Science*, 145:1–26, 1995.

[6] G. Cattaneo, A. Dennunzio, and L. Margara. Solution of some conjectures about topological properties of linear cellular automata. *Theoretical Computer Science*, 325:249–271, 2004.

[7] G. Cattaneo, E. Formenti, G. Manzini, and L. Margara. Ergodicity, transitivity, and regularity for linear cellular automata. *Theoretical Computer Science*, 233:147–164, 2000.

[8] J. Cervelle, A. Dennunzio, and E. Formenti. Chaotic behavior of cellular automata. In B. Meyers, editor, *Mathematical basis of cellular automata*, Encyclopedia of Complexity and System Science. Springer Verlag, 2008.

[9] K. Culik and S. Yu. Undecidability of cellular automata classification schemes. *Complex Systems*, 2:177–190, 1988.

[10] M. D'Amico, G. Manzini, and L. Margara. On computing the entropy of cellular automata. *Theoretical Computer Science*, 290:1629–1646, 2003.

[11] M. Delorme, E. Formenti, and J. Mazoyer. Open problems on cellular automata. Research Report RR2000-25, École Normale Supérieure de Lyon, june 2000.

[12] B. Durand, E. Formenti, and G. Varouchas. On undecidability of equicontinuity classification for cellular automata. volume AB of *Discrete Mathematics and Theorectical Computer Science*, pages 117–128, 2003.

[13] E. Formenti and P. Kůrka. Dynamics of cellular automata in non-compact spaces. In B. Meyers, editor, *Mathematical basis of cellular automata*, Encyclopedia of Complexity and System Science. Springer Verlag, 2008.

[14] R. H. Gilman. Classes of linear cellular automata. *Ergodic Theory & Dynamical Systems*, 7(105–118), 1987.

[15] M. Ito, N. Osato, and M. Nasu. Linear cellular automata over Z_m. *Journal of Computer and System sciences*, 27:125–140, 1983.

[16] J. Kari. Rice's theorem for the limit set of cellular automata. *Theoretical Computer Science*, 127(2):229–254, 1994.

[17] J. Kari. Tiling problem and undecidability in cellular automata. In B. Meyers, editor, *Mathematical basis of cellular automata*, Encyclopedia of Complexity and System Science. Springer Verlag, 2008.

[18] B. Kitchens. *Symbolic dynamics.* Universitext. Springer-Verlag, Berlin, 1998. One-sided, two-sided and countable state Markov shifts.

[19] P. Kůrka. Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory & Dynamical Systems*, 17:417–433, 1997.

[20] P. Kůrka. *Topological and Symbolic Dynamics.* Number 11 in Cours Spécialisés. Société Mathématique de France, Paris, 2004.

[21] P. Kůrka. Topological dynamics of one-dimensional cellular automata. In B. Meyers, editor, *Mathematical basis of cellular automata*, Encyclopedia of Complexity and System Science. Springer Verlag, 2008.

[22] P. Di Lena. Decidable properties for regular cellular automata. In *Fourth IFIP International Conference on Theoretical Computer Science—TCS 2006*, volume 209 of *IFIP Int. Fed. Inf. Process.*, pages 185–196. Springer, New York, 2006.

[23] P. Di Lena and L. Margara. Computational complexity of dynamical systems: the case of cellular automata. *Information and Computation*, 2008. To appear.

[24] Douglas Lind and Brian Marcus. *An introduction to symbolic dynamics and coding.* Cambridge University Press, Cambridge, 1995.

[25] G. Manzini and L. Margara. Invertible linear cellular automata over Z_m: Algorithmic and dynamical aspects. *Journal of Computer and System Sciences*, 1:60–67, 1998.

[26] G. Manzini and L. Margara. Attractors of D-dimensional linear cellular automata. *Journal of Computer and System Sciences*, 58(3):597–610, 1999.

[27] G. Manzini and L. Margara. A complete and efficiently computable topological classification of D-dimensional linear cellular automata over Z_m. *Theoretical Computer Science*, 221(1-2):157–177, 1999.

[28] J. Milnor. Directional entropies of cellular automaton maps. In *Disordered systems and biological organization*, number 20 in NATO Advanced Science Institutes Series F, pages 113–115. Springer, 1986.

[29] M. Nasu. Textile systems for endomorphisms and automorphisms of the shift. *Memoirs of the American Mathematical Society*, 114(546):viii+215, 1995.

[30] W. Parry. Symbolic dynamics and transformations of the unit interval. *Transactions of the American Mathematical Society*, 122:368–378, 1966.

[31] M. Pivato. The ergodic theory of cellular automata. In B. Meyers, editor, *Mathematical basis of cellular automata*, Encyclopedia of Complexity and System Science. Springer Verlag, 2008.

[32] M. Sablik. Directional dynamics for cellular automata. a sensitivity to the initial conditions approach. *Theoretical Computer Science*, 2008. To appear.

[33] T. Sato. Surjective linear cellular automata over Z_m. *Information Processing Letters*, 66(2):101–104, 1998.

Journées Automates Cellulaires 2008 (Uzès), pp. 54-64

SOFIC ONE HEAD MACHINES

A. GAJARDO

Departamento de Ingeniería Matemática, Universidad de Concepción,
Casilla 160-C, Concepción, Chile
E-mail address: `anahi@ing-mat.udec.cl`
URL: `http://www.ing-mat.udec.cl/~anahi`

Abstract. There are several systems consisting in an object that moves on the plane by following a given rule. It is frequently observed that these systems eventually fall into an unexplained repetitive movement. The general framework of k-dimensional Turing machines with only one head is adopted. A subshift is associated to each Turing machine, and its properties are studied. The subshift consists in the set of sequences of symbols that the machine reads together with the states that it has through each evolution. The focus is placed on the machines whose associated subshift is sofic. These machines cannot make long tours, i.e., the time between two consecutive visits to a given cell is bounded, and this property characterises them. It is proved that all of these machines eventually fall into a repetitive movement when starting over an initially periodic coloration. Nevertheless, it seems that the machines with a sofic subshift are too simple. Many known machines remain out of scope. As an example, the 0,1 and 2 pebble automata with 1 symbol are studied.

Introduction

We call "One Head Machine" an automaton that lives in a discrete space. It can walk, read and write symbols, and its behaviour is governed, at discrete time, by a deterministic and finitely described rule. Examples of this kind of dynamical systems are the Langton's Ant [9, 8, 6], the Pebble Automata [2], the one head Turing Machines [3, 7] and the Lorentz Lattice Gas [1, 10]. Such an object can represent a particle that collides with obstacles; a living being that interacts with its environment; an automaton that performs a task, etc.

All of these systems are more or less comprised by the following definition.

Definition 0.1. A *One Head Machine* over $\mathbb{Z}^k$ is a 4-tuple (S, Q, k, δ) where:

- S is a finite set, representing the state of the environment at each lattice point, and called *symbol set*,
- Q is a finite set, representing the internal state of the machine, called *state set*,
- $k \in \mathbb{N}$ represents the dimension of the lattice,

Key words and phrases: Multidimensional Turing Machine, Formal Language, Symbolic System.
This work has been supported by CONICYT FONDECYT #1061036.

- $\delta = (\delta_S, \delta_Q, \delta_D)$ is the *transition function*, where $\delta_i : S \times Q \to i$, for each $i = S$, Q or D, and $D = \{\pm e_j\}_{j=1}^{k}$ are the k canonical vectors in $\mathbb{Z}^k$ together with their opposites.

The elements of $\mathbb{Z}^k$ are called *cells*. A configuration of the system is given by an assignment of symbols to each cell, $c : \mathbb{Z}^k \to S$, called coloration; a position $g \in \mathbb{Z}^k$; and a state $q \in Q$, i.e., the phase space is $X = S^{\mathbb{Z}^k} \times \mathbb{Z}^k \times Q$.

The global transition function $T : X \to X$ is defined by $T((c, g, q)) = (c', g', q')$, where

- $q' = \delta_Q(c(g), q)$,
- $g' = g + \delta_D(c(g), q)$,
- $c'(g) = \delta_S(c(g), q)$ and $c'(u) = c(u)$ for all $u \neq g$.

This system can be fruitfully studied by projecting it into a symbolic system [4, 5, 7], as we precise in the next definition. This method works well due to a relevant feature of this system: all the changes happen only on the machine position, the rest of the coloration remaining static. Thus, if we register the sequence of symbols that the machine reads together with its state, we describe the entire evolution of the system without ambiguity.

Definition 0.2. Given a one head machine $M = (S, Q, k, \delta)$ and its associated dynamical system (X, T), let $\pi : X \to S \times Q$ be defined by $\pi(c, g, q) = (c(g), q)$ and let $\psi : X \to (S \times Q)^{\mathbb{N}}$ be defined by $\psi(x) = (\pi(T^n(x)))_{n \in \mathbb{N}}$. The *t-shift* of (X, T) is $S_T = \psi(X)$.

The set $\psi(X)$ represents all the possible sequences of pairs (symbol, state) that the machine can produce when considering all the possible initial configurations. Given an infinite sequence $y = \binom{\alpha_1 \alpha_2 \cdots}{q_1 q_2 \cdots} \in S_T$, we can deduce the machine itinerary. In fact, if we suppose that the initial position is 0, its position at iteration j must be:

$$I(y)_j = \sum_{i=1}^{j-1} \delta_D(\alpha_i, q_i) \quad (\forall 2 \leq j) \quad , \quad I(y)_1 = 0$$

and the set of visited cells is given by

$$V(y) = \{I(y)_j | 1 \leq j\}$$

The initial symbol of the visited cells can be deduced from y and it is given by the following formula

$$c_y(g) = \alpha_i \quad \text{where} \quad i = \min\{j | I(y)_j = g\} \quad (\forall g \in V(y))$$

The partial function c_y is a kind of pre-image of y by ψ in the following sense: if c is an extension of c_y to $\mathbb{Z}^k$ then $\psi(c, 0, q_1) = y$. This means that the sequence $\psi(x)$ contains information about the visited cells and discards the symbols of the other cells. Moreover, it is invariant under translations.

Remark 0.3. $I(y)$, $V(y)$ and c_y can be defined also if y is a finite word. In this context the following properties hold for every u, $v \in (S \times Q)^*$:

(1) $I(uv)_j = I(u)_j$, if $j \leq |u| + 1$.
(2) $I(uv)_j = I(u)_{|u|+1} + I(v)_{j-|u|}$, if $j \geq |u| + 1$.

The set S_T is sensitive to many of the machine properties. For example, if $\psi(x)$ is periodic we can deduce that the sequence of movements of the machine is periodic, i.e., that the machine is making a regular movement (which can be propagative or cyclic).

Next section recall some concepts from symbolic dynamics and presents basic properties of S_T. In Section 2, we characterise the machines having a sofic t-shift and we prove an important feature of these systems: starting over a periodic coloration with a finite number of perturbations the machine always finishes by falling in a periodic movement. The last section shows how to adapt the this theory to a particular type of pebble automaton, we obtain two results already proved by Delorme and Mazoyer [2].

1. Basic notions and previous results

Given a finite set Σ, the set $\Sigma^{\mathbb{N}}$ denotes the set of infinite sequences of elements of Σ. A function σ is defined on $\Sigma^{\mathbb{N}}$ by: $\sigma(y_1y_2y_3...) = y_2y_3y_4...$, it is called the *shift* function. A metric can be defined on $\Sigma^{\mathbb{N}}$ by: $d(y,z) = 2^{-n}$, where n is the smallest index such that $y_n \neq z_n$. This metric makes $\Sigma^{\mathbb{N}}$ compact and σ continuous. Closed and σ invariant sets are called *subshifts.*

The finite sequences of elements of Σ are called *words.* The set of words is denoted by Σ^*. If a word $v \in \Sigma^*$ appears as a subsequence of an infinite sequence $y \in \Sigma^{\mathbb{N}}$, it is called a *factor* of y, and this is denoted by $v \sqsubseteq y$. A *language* is any subset of Σ^*.

Any subshift $Y \subset \Sigma^{\mathbb{N}}$ has an associated language $L(Y) \subset \Sigma^*$, defined by:

$$L(Y) = \{w \in \Sigma^* : (\exists y \in Y)\ w \sqsubseteq y\}.$$

This is the *factors language* of Y. The factors language characterises Y because $Y = \{y \in A^{\mathbb{N}} : (\forall u \sqsubseteq y)\ u \in L(Y)\}$.

Another way to characterise a subshift is through a set of forbidden words. A language P is a *set of forbidden words* for Y if

$$Y = \{y \in A^{\mathbb{N}} : (\forall u \sqsubseteq y)\ u \notin P\}.$$

If Y has a finite set of forbidden words, Y is said to be a *shift of finite type (SFT).*

The complexity of Y is defined with reference to the complexity of its language $L(Y)$. For instance, Y is said to be *sofic* if $L(Y)$ is regular. It is easy to see that if Y is a SFT, Y is also sofic.

Let us come back to the shift associated to a one head machine: S_T. It is a subshift. Moreover, functions T, σ and ψ satisfy $\psi \circ T = \sigma \circ \psi$. The following result characterises the words in $L(S_T)$. This property can be easily proved by induction.

Lemma 1.1. [7] *If $w = \binom{\alpha_1..\alpha_n}{q_1..q_n} \in L(S_T)$, then for all $i \in \{1,..,n\}$:*

$$q_{i+1} = \delta_Q(\alpha_i, q_i) \qquad \textit{(state coherence)} \qquad (1.1)$$

and for any pair $1 \leq i < j \leq n$, such that $I(w)_i = I(w)_j$ (say $= g$) and for every k between i and j, $I(w)_k \neq g$, one has that

$$\alpha_j = \delta_S(\alpha_i, q_i). \qquad \textit{(writing coherence)} \qquad (1.2)$$

Moreover, these are sufficient conditions for w to belong to $L(S_T)$.

Equation (1.1) expresses that the sequence of states must be coherent with the transition rule of the machine. Equation (1.2) expresses that when the machine visits a cell a second time, it must find the symbol that it wrote there when it visited it the first time. A set of forbidden words for S_T can be obtained from these two equations. From Equation (1.1) we obtain the following set:

$$P_1 = \left\{ \begin{pmatrix} \alpha\beta \\ qp \end{pmatrix} : p \neq \delta_Q(\alpha, q) \right\}$$

Equation (1.2) refers to trajectories that visit two times the same cell.

Definition 1.2. A word $w \in L(S_T)$ whose itinerary starts and finishes in the same cell and does not visit that cell in between, is called a *cycle*, i.e., w is a cycle if it satisfies: $I(w)_1 = I(w)_{|w|+1} = 0$ and $I(w)_j \neq 0$, for all $j \in \{2, .., |w|\}$. When saying "the cycle w" we will being making reference to either the word w, the itinerary $I(w)$ or the set $V(w)$; the interpretation will be clear from the context.

There is a set of forbidden words for each cycle $w = \binom{\alpha_1..\alpha_n}{q_1..q_n}$: the word $\binom{\alpha_1..\alpha_n\beta}{q_1..q_nq}$ is forbidden for every $\beta \neq \delta_S(\alpha_1, q_1)$. Thus, we obtain the following set of forbidden words:

$$P_2 = \left\{ \begin{pmatrix} \alpha_1..\alpha_n\beta \\ q_1..q_nq \end{pmatrix} \; : \; \begin{pmatrix} \alpha_1..\alpha_n \\ q_1..q_n \end{pmatrix} \text{ is a cycle and } \beta \neq \delta_S(\alpha_1, q_1) \right\}.$$

S_T is defined by the set of forbidden words $P = P_1 \cup P_2$. P_1 is finite but P_2 may be infinite, depending on the behaviour of the machine. For example, if the machine never visits a cell more than once, P_2 is empty. If the number of cycles is finite, P_2 is finite. In both cases, the set of forbidden words of S_T is finite and therefore S_T is a SFT. With some work it is possible to prove the converse [7], i.e., if S_T is a SFT, then the number of cycles is finite. If S_T is a SFT, the machine has significant movement restrictions that prevent it from making long cycles. What happens when S_T is sofic? In [7] the one dimensional case was studied and it was established that if S_T is sofic, then it is of finite type too. Is this also true in Z^k? What is the relation between the complexity of S_T and the machine behaviour? In order to answer these questions we need more information about the relation between S_T, T and the automaton that recognises S_T. Let us recall some definitions.

Definition 1.3. A Deterministic Finite Automaton (DFA) is a 5-tuple $M = (A, \Omega, \lambda, o_0, F)$ where A is the *input alphabet, Ω is the* states set, $\lambda : A \times \Omega \to \Omega$ is a partial function called *transition function*, $o_0 \in \Omega$ is the *initial state and F is the set of* final states.

A labelled graph, G_M, is associated to M. Its set of vertices is Ω, and the label of an edge (e, f) is 'a' if and only if $f = \lambda(a, e)$.

The language recognised by M consists of all words w in A^* such that there exists a path in G_M with label w, starting on vertex o_0 and finishing on a vertex $f \in F$.

If a language L is the factors language of some subshift, then it is closed for the factor relation, i.e., if $w \in L$ and u is a subword of w, then $u \in L$. Consequently, the automaton $M = (S \times Q, \Omega, \lambda, o_0, F)$ that recognises it can be chosen such that $F = \Omega$. In the following we will omit the set F from the automata definition.

Definition 1.4. A language is said to be *regular if it is recognised by some DFA.*

Remark 1.5. Given a vertex $\nu \in \Omega$, it holds that:

(1) If ν has an input edge labelled by (α, q), then all the exiting edges of ν have a label of the form $(\beta, \delta_Q(\alpha, q))$, with $\beta \in S$, because the next state of the machine is uniquely determined by α and q and it is $\delta_Q(\alpha, q)$.

(2) Since every word in $L(S_T)$ defines a unique path in G_M, Equation (1.2) implies that ν has only one exiting edge if and only if every path from o_0 to ν corresponds to an itinerary that has already visited its last cell. Otherwise, ν has exactly $|S|$ exiting edges.

(3) The last assertion is not valid when $|S| = 1$. But, in this case, S_T is of finite type, more precisely, it is a finite set composed by eventually periodic sequences. The cycles exist in a finite quantity. The vertices of the automaton that recognises it have degree 1 (except for o_0)

This remark allows us to characterise the one head machines whose t-shift is sofic.

2. When S_T is sofic

Let us consider a metric in $\mathbb{Z}^k$ as follows. Given two points: $p = (p_1, p_2, .., p_k)$ and $g = (g_1, g_2, .., g_k)$, the distance between p and g is

$$d(p, g) = \sum_{i=1}^{k} |p_i - g_i|.$$

Thus the set $B(p, n) = \{g \in \mathbb{Z}^k : d(p, g) \leq n\}$ represents the ball of radius n and centre p.

Lemma 2.1. *The number of cycles of $L(S_T)$ is finite if and only if the distance that a cycle can attain form the origin is bounded.*

Proof. Let us suppose that every cycle is contained in a ball of radius n. The number of configurations defined in this ball is $|S|^{2n^k} \times 2n^k \times |Q|$. Each configuration corresponds to at most one cycle, hence there is a finite number of cycles. ■

The following theorem has already been proved for $k = 1$ in [7].

Theorem 2.2. *S_T is sofic if and only if the number of cycles of $L(S_T)$ is finite.*

Proof. In one direction, the result is trivial since every SFT is sofic.

Let us suppose that S_T is sofic. Therefore, there exists a DFA $M = (S \times Q, \Omega, \lambda, o_0)$ that recognises it and satisfies the conditions given in Remark 1.5. Let us suppose that the length of the cycles is arbitrary large. Lemma 2.1 implies that for every natural n there is a cycle that attains a distance bigger than n from its initial cell: 0. Let us consider the set of cycles that makes this for $n = |\Omega|$. Let us choose from this set the shortest cycle $w = w_1..w_m = \binom{\alpha_1..\alpha_m}{q_1..q_m}$.

Given $r < m$, we can use Remark 0.3, with $u = w_1..w_{r-1}$ and $v = w_r..w_m$ and the fact that $I(w)_{|w|+1} = 0$ to obtain that:

$$I(w)_r = -I(w_r..w_m)_{m-r+1} \neq 0. \tag{2.1}$$

The cycle w corresponds to a unique path in the graph G_M: $o_0 o_1, .., o_m$. From Remark 1.5, the vertex o_m has an exit degree equal to 1, because the last cell of w (cell 0) has already been visited.

Let $l \leq m$ be such that $d(I(w)_l, 0) > n$. Since for every j, $d(I(w)_j, 0) \leq j - 1$, we can assert that $l - 1 > n$. In consequence, there must exist two repeated vertices between o_1 and o_{l-1}: $o_i = o_j$. Thus $o_0 o_1..o_i o_{j+1}..o_m$ is also a path in G_M, its label is $u = w_1 w_2..w_i w_{j+1}..w_m$, its length is $t = m - j + i$, and $I(u)_{t+1}$ has already been visited. This implies that there exists r such that $I(u)_r = I(u)_{t+1}$. By using Remark 0.3 over u decomposed by $u_1..r_{r-1}$ and $u_r..u_t$, we obtain that $I(u_r..u_t)_{t-r+2} = 0$, i.e., $u_r..u_t$ is a cycle of length $t - r + 1$.

If $r > i$, $u_r..u_t = w_{r+j-i}..w_m$ which, from Equation (2.1), is not a cycle, therefore $r \leq i$. In summary, $1 \leq r \leq i < j < l < m$. Now, we use Remark 0.3 again over $u_r..u_{k-1}$ and $u_k..u_t$, with $k = l - j + i$, and we obtain

$$\begin{aligned} I(u_r..u_{k-1})_{k-r+1} &= -I(u_k..u_t)_{t-k+1} \\ &= -I(w_l..w_m)_{m-l+1}, && \text{since } k > i, \\ &= I(w)_l, && \text{due to Equation (2.1).} \end{aligned}$$

Hence $d(I(u_r..u_t)_{k-r+1}, 0) = d(I(w)_l, 0) > n$. We conclude that $u_r..u_t$ is a cycle that attains a distance bigger than n and is shorter than w, which is a contradiction. ∎

This implies that the machines whose t-shift is sofic are very simple: they cannot revisit far cells; the diversity of cycles they can do is finite; its t-shift is also a SFT; and all the closed itineraries can be putted inside a finite ball. Since the state of the visited cells can be interpreted as the external "remembers" of the machine, this ball represents its attainable memory. Sofic machines have in fact a finite memory.

Corollary 2.3. *If S_T is sofic and $V(\psi(c, g, q))$ is infinite, then the number of times that the machine visits each cell is bounded by a finite constant which only depends on the machine.*

Proof. Let r be the radius of a ball containing all the closed trajectories. The number of configurations defined on this ball is finite, say N. Let $x = (c, g, q)$ be an initial configuration. If some cell p is visited more than N times during the evolution of T on x, we can assert that the machine remained inside the ball of radius r and centre p from the first to the last time that it visited p. Within this time, some configuration of the ball has appeared two or more times. Which means that the system has fallen in a periodic point and that its complete itinerary is contained in a finite set. ∎

When a sofic machine starts over a periodic coloration, its behaviour is particularly simple, as the following theorem establishes. This theorem is proved in [7] for $k = 1$.

Theorem 2.4. *If S_T is sofic and c is periodic except for a finite number of cells, then $\psi(c, g, q)$ is eventually periodic for every $g \in \mathbb{Z}^k$ and $q \in Q$.*

Proof. We can suppose, without loss of generality, that the initial position of the machine is 0. Let q_0 be its initial internal state, and $c : \mathbb{Z}^k \to S$ a periodic coloration except for a finite number of cells. This means that c is equal to some periodic coloration d except for a finite set of cells E.

Now, let us study $y = \binom{\alpha_i}{q_i}_{i\in\mathbb{N}} = \psi(c, 0, q_0)$. Two possibilities appear: $V(y)$ can be finite or not. If it is finite, $(T^i(c, 0, q_0))_{i\in\mathbb{N}}$ is eventually periodic, and so is y.

Let us analyse the case when $V(y)$ is infinite. Let n be the last iteration in which a cell of E is visited. This means that after iteration n every cell g either has been already visited or its state is given by $d(g)$.

Coloration d consists in the repetition of a pattern defined on a rectangle R. Let us assume that $R = \{0, .., r_1\} \times \{0, .., r_2\} \times \cdots \times \{0, .., r_k\}$, where $r_i \in \mathbb{N}$ for all i. This means that the value of d at cell $g = (g_1, g_2, .., g_k)$ is equal to $d(g_1 \text{mod } r_1, g_2 \text{mod } r_2, .., g_k \text{mod } r_k)$.

Now let $M = (S \times Q, \Omega, \lambda : (S \times Q) \times \Omega \to \Omega, o_0)$ be the automaton that recognises S_T. We define $\overline{M} = (S \times Q, \Omega \times R, \overline{\lambda}, (o_0, 0))$ where $\overline{\lambda} : (S \times Q) \times (\Omega \times R) \to (\Omega \times R)$ is defined by:

$\overline{\lambda}(\binom{\alpha}{q}, \binom{\mu}{f}) = \binom{\nu}{g} \Leftrightarrow \lambda(\binom{\alpha}{q}, \mu) = \nu$ and $(\forall i)\ f_i = (g_i + \delta_D(\alpha, q)_i) \text{ mod } r_i$.

$\overline{M}$ recognises the same language than M. Moreover, it registers the position of the machine modulo R, i.e., if $((o_i, g^i))_{i\in\mathbb{N}}$ is the sequence of vertices in $G_{\overline{M}}$ whose label is y, then, for every $j \in \mathbb{N}$, $I(y)_{j+1} = g^j$ modulo R.

From Remark 1.5, we distinguish two kinds of vertices in $G_{\overline{M}}$: either $deg((o_i, g^i)) = 1$ or $deg((o_i, g^i)) = |S|$. If $deg((o_i, g^i)) = |S|$, we know that $I(y)_{i+1}$ is being visited by the first time at iteration $i + 1$. If in addition $i > n$, we can assert that $c(I(y)_{i+1}) = d(I(y)_{i+1}) = d(g^i)$. Consequently, the label of the arc $((o_i, g^i), (o_{i+1}, g^{i+1}))$ is $d(g^i)$, this means that (o_{i+1}, g^{i+1}) is uniquely determined by (o_i, g^i). Thus starting from (o_n, g^n) only one sequence of vertices of $G_{\overline{M}}$ can be taken, hence $((o_i, g^i))_{i\in\mathbb{N}}$ is ultimately periodic and so is y. ■

These two theorems can be easily proved for other regular grids than $\mathbb{Z}^k$, for example, Cayley graphs of groups.

3. Pebble automata with one symbol

Pebble automata are two dimensional one head machines that cannot write but are provided with a set of "pebbles" that they can drop and recover in order to mark their way. They could be seen as a particular kind of one head machine by assimilating the pebbles as part of the internal state and space symbols. The following definition is adapted from [2] for the particular case where the space has only one symbol.

Definition 3.1. A *Pebble Automata* is a 3-tuple (Q, δ, l) where:

- Q is a finite set, representing the internal state of the machine,
- l represents the number of pebbles, and
- $\delta = (\delta_P, \delta_Q, \delta_D)$ is the transition function, where
  - $\delta_P : Q \times \{0,1\}^l \times \{0,1\}^l \to \{0,1\}^l$ determines the pebbles that will be taken or dropped,
  - $\delta_Q : Q \times \{0,1\}^l \times \{0,1\}^l \to Q$ determines the new machine state, and
  - $\delta_D : Q \times \{0,1\}^l \times \{0,1\}^l \to D$ determines the moving direction of the machine.

Moreover, δ_P satisfies that for every $q \in Q$ and $p, r \in \{0,1\}^l$, $\delta_P(q, p, r) \le p + r$; this assures that the machine can only act over pebbles that the machine is carrying or that are on the current machine position.

The configuration of the system is given by an assignment of pebbles to each cell $c : \mathbb{Z}^2 \to \{0,1\}^l$, a position $g \in \mathbb{Z}^2$, a state $q \in Q$, and the pebbles reserve of the machine $p \in \{0,1\}^l$. Thus, the phase space is $X = (\{0,1\}^l)^{\mathbb{Z}^2} \times \mathbb{Z}^2 \times Q \times \{0,1\}^l$.

The global transition function $A : X \to X$ is defined by $A((c, g, q, p)) = (c', g', q', p')$, where

- $q' = \delta_Q(q, c(g), p)$,
- $p' = (p + \delta_P(q, c(g), p)) \bmod 2$,
- $g' = g + \delta_D(q, c(g), p)$,
- $c'(g) = (c(g) + \delta_P(q, c(g), p)) \bmod 2$ and for all $u \ne g$, $c'(u) = c(u)$.[1]

The system starts with the empty configuration $c(u) = (0, 0, .., 0)$ and all the pebbles on the machine: $p = (1, 1, .., 1)$.

[1]let us remark that $c(g) + p = c'(g) + p'$.

In [2], the behaviour of these automata is studied for the case in which the number of symbols is 1. The object of doing this is to analyse the ability of the automaton to explore the plane without external help. They prove that this task is impossible when the automaton has less than 3 pebbles. Delorme and Mazoyer prove this by showing that the d-pebble automata have serious restrictions on their movements when $d \leq 2$. This can be illustrated within the present theory too. 0-pebble automata are a trivial case. Since they have no pebble, they are actually one head machines where $|S| = 1$, and $k = 2$. Thus the number of cycles is bounded and, independently from the initial state $q_0 \in Q$, the machine will always finish by making repetitive movements.

3.1. 1-pebble automata

As we said before, pebble automata can be seen as a particular case of one head machine by assimilating the pebbles as part of the internal state of the machine and symbols of the space. This means to define a one head machine $M = (\overline{S}, \overline{Q}, 2, \overline{\delta})$ where $\overline{Q} = Q \times \{0,1\}$, $\overline{S} = \{0,1\}$ and $\overline{\delta}$ is defined appropriately. The second component of the state represents the pebble reserve of the machine. The symbol represents the pebble content of the cells.

The difference between pebble automata and one head machines is that in the pebble automata the total number of pebbles in the system is fixed and constant. A 1-pebble automaton is a one head machine that works over a particular set of configurations: those with exactly one pebble in the whole space.

Thus the shift of a pebble automata is smaller than the shift of its corresponding one-head machine. In the pebble automata, not only cycles define forbidden words. If the pebble automata put the pebble somewhere, it cannot find it elsewhere. This fact induces additional forbidden words. First, P_0 forbids to find the pebble in the plane when the pebble is on the machine.

$$P_0 = \left\{ \begin{pmatrix} \alpha \\ \overline{q} \end{pmatrix} : \overline{q} = (q,1) \wedge \alpha = 1 \right\}.$$

Second, let $w = \binom{\alpha_1..\alpha_n}{\overline{q}_1..\overline{q}_n} \in (\{0,1\} \times \overline{Q})^*$ be such that $I(w)_i \neq 0$ for every $i \geq 2$. In this case, we know that if at the beginning of w the initial cell contains the pebble, i.e., $\alpha_1 = 1$, and the machine does not take it ($\delta_P(\alpha_1, q_1) = 0$), then $\alpha_i = 0$ for all $i \geq 2$. The same happens if the machine has the pebble and drops it at the beginning, i.e., if $\alpha_1 = 0 \wedge \delta_P(\alpha_1, q_1) = 1$. In these cases, we can add the forbidden word $\binom{\alpha_1..\alpha_n 1}{\overline{q}_1..\overline{q}_n\overline{q}_{n+1}}$. This defines the following set of forbidden words:

$$P_3' = \left\{ w = \begin{pmatrix} \alpha_1..\alpha_n 1 \\ \overline{q}_1..\overline{q}_n\overline{q}_{n+1} \end{pmatrix} : \alpha_1 + \delta_P(\alpha_1, q_1) = 1 \wedge (\forall i \geq 2)\ I(w)_i \neq 0 \right\}.$$

We only need to consider a finite part of P_3', because if the machine has no pebble, it behaves like a 0-pebble automaton where the length of cycles is bounded, say by M, and therefore longer trajectories do not visit 0. Thus forbidden words longer than M can be replaced by the word $\binom{00..01}{\overline{q}_1..\overline{q}_M}$. We obtain a new set of forbidden words:

$$\begin{aligned} P_3 &= \left\{ w = \begin{pmatrix} \alpha_1..\alpha_n 1 \\ \overline{q}_1..\overline{q}_n\overline{q}_{n+1} \end{pmatrix} : \alpha_1 + \delta_P(\alpha_1, q_1) = 1 \wedge (\forall i \geq 2)\ I(w)_i \neq 0 \wedge n \leq M \right\} \\ &\bigcup \left\{ \begin{pmatrix} 00..01 \\ \overline{q}_1..\overline{q}_M \end{pmatrix} : \overline{q}_1 = (q_1, 0) \right\}. \end{aligned}$$

Finally, each time we have a word $w = \binom{0..01}{\overline{q}_1..\overline{q}_n}$ where $\overline{q}_1 = (q_1, 0)$, we know that this pebble must have been dropped by the automaton at some past iteration, then w is a suffix of a cycle that begins by leaving the pebble on the plane. We thus obtain the forbidden set P'_4.

$$P'_4 = \left\{ \begin{pmatrix} 0..01 \\ \overline{q}_1..\overline{q}_n \end{pmatrix} : \overline{q}_1 = (q_1, 0) \wedge (\forall v \in C_0) \begin{pmatrix} 0..0 \\ \overline{q}_1..\overline{q}_{n-1} \end{pmatrix} \not\preccurlyeq v \right\}.$$

Where C_0 denotes the set of cycles that begins by leaving the pebble and $\preccurlyeq$ is the suffix relation. Again, only a finite part of P'_4 is enough, because we know that every word longer than M is not suffix of some cycle. Hence, we consider the set P_4:

$$P_4 = \left\{ \begin{pmatrix} 0..01 \\ \overline{q}_1..\overline{q}_n \end{pmatrix} : \overline{q}_1 = (q_1, 0) \wedge (\forall v \in C_0) \begin{pmatrix} 0..0 \\ \overline{q}_1..\overline{q}_{n-1} \end{pmatrix} \not\preccurlyeq v \wedge n \leq M \right\}$$

By avoiding these words, we assure that the pebble is always found exactly where it was dropped.

Let us recall now the definition of the set P_2:

$$P_2 = \left\{ \begin{pmatrix} \alpha_1..\alpha_n\beta \\ \overline{q}_1..\overline{q}_n\overline{q} \end{pmatrix} : \begin{pmatrix} \alpha_1..\alpha_n \\ \overline{q}_1..\overline{q}_n \end{pmatrix} \text{ is a cycle and } \beta \neq \alpha_1 + \delta_P(\alpha_1, \overline{q}_1) \text{mod } 2 \right\}$$

We distinguish three types of cycles, depending on the position of the pebble at the beginning of the itinerary: a) the pebble is left at 0 ($\alpha_1 + \delta_P(\alpha_1, \overline{q}_1) = 1$), b) the pebble is not at 0 nor on the machine ($\alpha_1 = 0$ and $\overline{q}_1 = (q, 0)$, $q \in Q$), c) the pebble is carried by the machine ($(\alpha_1 + \delta_P(\alpha_1, \overline{q}_1) = 0 \wedge \overline{q}_1 = (q, 1)) \vee \alpha_1 + \delta_P(\alpha_1, \overline{q}_1) = 2$).

There is only a finite number of cycles of type a). If the cycle is of type b), there are two cases. First, the pebble does not appear in the cycle, it behaves like a 0-pebble automaton, therefore there is only a finite quantity of these cycles. Second, the pebble is found during the cycle. In this case, by avoiding the forbidden words of P_3 and P_0 we prevent from finding the pebble at position 0. Finally, if the cycle is of type c), two cases appear again. First, the pebble is over the machine during the whole cycle; in this case, the set P_0 assures that the pebble will not be found at the initial cell. Second, the machine drops the pebble somewhere; in this case, by avoiding the words of P_3 we preclude the possibility of finding the pebble at 0. It follows that, only a finite number of words of P_2 are necessary. The union of P_0, P_1, P_3, P_4 and a finite part of P_2 is a forbidden set for S_A. We conclude that S_A is of finite type.

Remark 1.5 is not valid in this case because we are using information that is not available for general one head machines. But Remark 1.5 is used in Theorem 2.4 to have vertices with exit degree equal to one. In $L(S_A)$ only short words that do not contain the pebble can be enlarged in two different ways, because words longer than M that has no the pebble will never have it; and if a word contain the pebble, its future is uniquely determined. This means that after iteration M, all the attainable vertices has degree 1. Thus the proof of Theorem 2.4 applies to the present case and we can conclude that the automaton will always finish by making a repetitive movement.

3.2. 2-pebble automata

In these automata, trajectories as those of Figure 1 can occur, as it is shown in [2]. Theorem 2.2 can be used to assert that, in this case, S_A is not sofic. We can wonder about the complexity of S_A. Delorme-Mazoyer proved that these automata cannot explore the plane, this suggest that its complexity cannot be very high.

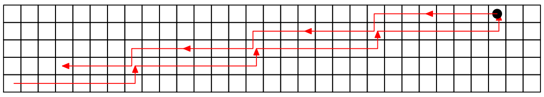


Figure 1: A trajectory that may contain cycles of arbitrary length. The automaton comes from below, it finds the pebble on top and comes back.

4. Discussion

The results of this paper apply to very simple machines. Sometimes, proving the hypothesis of our theorems is not easy. Nevertheless, we think that the present paper is a first step in the construction of effective tools to understand the dynamics of one head machines.

The study of pebble automaton was not easy, the principal reason is that pebble automaton do not feet with the definition of one head machine. Delorme and Mazoyer obtained the same results through an analysis of similar complexity

The next step may be to study machines with a context free language. In such a machine, the cycles may be very long, but they probably have a simple structure. Maybe the 2-pebble automata with one symbol is in this class. It would be important to study the dynamical properties of these machines.

Langton's Ant is a one head machine that, apparently, always falls in a periodic movement when it starts with a coloration with a finite number of black cells. Unhappily the cycles that Langton's Ant can do can be very complicated and arbitrarily long, hence our results do not apply. But maybe the ideas presented in the proofs may help to understand Langton's Ant behaviour.

Acknowledgement

The author wishes to acknowledge the referees for their careful reading and their deep comments.

References

[1] L. A. Bunimovich and S. Troubetzkoy. Topological dynamics of flipping Lorentz lattice gas models. *J. of Stat. Physics*, 72:297–307, 1993.

[2] M. Delorme and J. Mazoyer. Pebble automata. figures families recognition and universality. Technical Report 32, Ecole Normale Supérieure de Lyon, Lyon,France, 1999.

[3] A. K. Dewdney. Computer recreations: Two-dimensional Turing machines and tur-mites make tracks on a plane. *Scientific American*, pages 124–127, September 1989.

[4] A. Gajardo. *Dependence of the behavior of the dynamical system Langton's ant on the network topology.* PhD thesis, Universidad de Chile and École normale supérieure de Lyon, 2001.

[5] A. Gajardo. A symbolic projection of Langton's ant (extended abstract). In *Proc. (Discrete Models for Complex Systems) (DM-CS'03)*, volume AB, pages 57–68, 2003.

[6] A. Gajardo and E. Goles. Dynamics of a class of ants on a one-dimensional lattice. *Theor. Comput. Sci.*, 322(2):267–283, 2004.

[7] A. Gajardo and J. Mazoyer. One head machines from a symbolic approach. *Theor. Comput. Sci.*, 370:34–47, 2007.

[8] A. Gajardo, A. Moreira, and E. Goles. Complexity of Langton's ant. *Discrete Applied Mathematics*, 117:41–50, 2002.

[9] C. G. Langton. Studying artificial life with cellular automata. *Physica D*, 22:120–149, 1986.

[10] A. Quas. Infinite paths in a Lorentz lattice gas model. *Probab. Theory Rel*, 114(2):229–244, 1999.

Journées Automates Cellulaires 2008 (Uzès), pp. 65-73

A PARTICLE DISPLACEMENT REPRESENTATION FOR CONSERVATION LAWS IN TWO-DIMENSIONAL CELLULAR AUTOMATA

JARKKO KARI [1] AND SIAMAK TAATI [2]

[1] Department of Mathematics, University of Turku, FI-20014 Turku, Finland
E-mail address: `jkari@utu.fi`
URL: `http://users.utu.fi/jkari/`

[2] Turku Centre for Computer Science, FI-20520 Turku, Finland
E-mail address: `staati@utu.fi`
URL: `http://users.utu.fi/staati/`

ABSTRACT. The problem of describing the dynamics of a conserved energy in a cellular automaton in terms of local movements of “particles” (quanta of that energy) has attracted some people’s attention. The one-dimensional case was already solved by Fukś (2000) and Pivato (2002). For the two-dimensional cellular automata, we show that every (context-free) conservation law can be expressed in terms of such particle displacements.

Introduction

Let $\mathbb{L} = \mathbb{Z}^d$ be the d-dimensional square *lattice*, and S a finite set of *states*. Every cellular automaton (CA for short) $F : S^{\mathbb{L}} \to S^{\mathbb{L}}$ maps the uniform configurations into the uniform configurations. Two configurations $x, y : \mathbb{L} \to S$ are *asymptotic* if they agree on all but finitely many cells of the lattice. The image of asymptotic configurations under every cellular automaton remain asymptotic.

A (context-free) *energy* assignment is a function $\mu : S \to \mathbb{R}$. The *μ-content* of a finite pattern $p : A \to S$ ($A \subseteq \mathbb{L}$ finite) is the sum $M(p) \triangleq \sum_{i\in A} \mu(p[i])$. For every two asymptotic configurations $x, y \in S^{\mathbb{L}}$, the corresponding *energy difference* is

$$\delta M(x, y) \triangleq \sum_{i\in\mathbb{L}} [\mu(y[i]) - \mu(x[i])] \tag{0.1}$$

which is clearly well-defined (only a finite number of terms are non-zero). The energy μ is *conserved* by a cellular automaton $F : S^{\mathbb{L}} \to S^{\mathbb{L}}$, if

$$\delta M(Fx, Fy) = \delta M(x, y) \tag{0.2}$$

Key words and phrases: Conservation Laws, Flows, Particles.
Research supported by the Academy of Finland grant 211967.

for every two asymptotic configurations x and y. This is equivalent to the formulations in terms of finite or periodic configurations [7, 3]. In particular, one can show that, if F maps an a-uniform configuration to a b-uniform configuration, we must have $\mu(a) = \mu(b)$.

For a conserved energy μ, it is desirable to find a local rule that explains the microscopic dynamics of μ under the iteration of F, in terms of "flows" of energy from one cell to another. More specifically, a *flow* for μ is a mapping $x, i, j \mapsto \Phi_{i\to j}(x) \in \mathbb{R}$ for $x \in S^{\mathbb{L}}$ and $i, j \in \mathbb{L}$ that satisfies the following conditions:

a) For every configuration x and every cell a,

$$\mu\left(x[a]\right) = \sum_{j\in\mathbb{L}} \Phi_{a\to j}(x)\,, \tag{0.3}$$

b) For every configuration x and every cell a,

$$\sum_{i\in\mathbb{L}} \Phi_{i\to a}(x) = \mu\left((Fx)[a]\right)\,, \tag{0.4}$$

c) There exist finite sets $K, I \subseteq \mathbb{L}$, and a rule $\varphi : S^K \times I \to \mathbb{R}$ such that,

$$\Phi_{i\to j}(x) = \begin{cases} \varphi\left(x[j+K], i-j\right) & \text{if } i-j \in I, \\ 0 & \text{otherwise,} \end{cases} \tag{0.5}$$

for every $x \in S^{\mathbb{L}}$ and $i, j \in \mathbb{L}$.

Here, $f[A]$ denotes the restriction of a function f to a subset A of its domain. Equations (0.4) and (0.3) are called the *continuity equations.* Equation (0.5) states that the amount of the flows toward each cell is decided locally, by looking at a finite *neighborhood* K of that cell. The set I is the set of *directions* from which energy flows into a cell. The local rule φ is called an *inflow.* An energy μ is *locally conserved* by F, if it has an inflow.

Proposition 0.1 (Hattori and Takesue [7]). *In cellular automata, conserved energies are locally conserved.*

We remark that the third condition in the above definition could equivalently be formulated in terms of an *outflow.*

Recently a number of people have shown interest in flows that can be interpreted as displacement of "particles" (see e.g. [6, 10, 9, 1]). In such a case, the μ-content of a pattern p is seen as the sum of the energies (or masses) of the particles in p.

Recall that every finitely generated subgroup of $\mathbb{R}$ is isomorphic to $\mathbb{Z}^m$ for some $m \geq 0$. If an energy $\mu : S \to \mathbb{Z}^m$ is conserved by a cellular automaton F, each of its m components must be conserved independently. If we are able to find "particle flows" for each component, we can extend our interpretation for μ by assuming m different types of particles which flow independently. So, without loss of generality, we can concentrate on the case $\mu : S \to \mathbb{Z}$ or $\mu : S \to \mathbb{Q}$. Now, if the particles are indistinguishable and each have the same energy $\varepsilon \in \mathbb{R}$, for every state $s \in S$ we must have $\mu(s) = \nu(s)\cdot\varepsilon$, where $\nu(s) \in \mathbb{Z}^{\geq 0}$ is the number of particles in s. Thus, it makes sense to assume μ is everywhere non-negative (or everywhere non-positive).

Formally, let $\mu : S \to \mathbb{Q}^{\geq 0}$ be a conserved energy for a cellular automaton F. A *particle flow* for μ is a flow Φ whose values are from non-negative rationals $Q^{\geq 0}$. Let Φ be a particle flow which is defined by an inflow $\varphi : S^K \times I \to \mathbb{Q}^{\geq 0}$. Let $\varepsilon > 0$ be such that $\varphi(p, i)/\varepsilon \in \mathbb{Z}^{\geq 0}$ for every p and i. Then ε is the *μ-content of a particle* and the function $\rho(\cdot,\cdot) \triangleq \varphi(\cdot,\cdot)/\varepsilon$

is called a *particle displacement rule*. For every state $s \in S$, $\nu(s) \triangleq \mu(s)/\varepsilon$ is the *number of particles* in s.

Proposition 0.2 (Fukś [6] and Pivato [10])**.** *Let $F : S^{\mathbb{Z}} \to S^{\mathbb{Z}}$ be a one-dimensional cellular automaton, and $\mu : S \to \mathbb{N}$ an energy conserved by F. Then, μ has a particle flow.*

We extend this result to the two-dimensional CA. In Section 1, we show how to construct a particle flow for a conserved energy in a two-dimensional radius-$\frac{1}{2}$ CA. In Section 2, we give a sketch of how this can be exploited in the case of arbitrary neighborhoods. Some open problems are proposed in Section 3.

1. Particle Flows in Radius One Half CA

Let $F : S^{\mathbb{Z}^2} \to S^{\mathbb{Z}^2}$ be a two-dimensional CA with neighborhood

$$N = \{(0,0),(0,1),(1,1),(1,0)\} \tag{1.1}$$

and local rule $f : S^N \to S$. The neighbors $(0,0)$, $(0,1)$, $(1,1)$ and $(1,0)$ are interpreted, respectively, as the down-left (`dl`), up-left (`ul`), up-right (`ur`) and down-right (`dr`) neighbors. Such a neighborhood is often called *radius-*$\frac{1}{2}$.

To simplify our exposition, let us distinguish between *neighbors* of a cell i, and the cells *adjacent* to it. The former are the cells $i+\texttt{dl}$, $i+\texttt{ul}$, $i+\texttt{ur}$ and $i+\texttt{dr}$ one step before, while the latter are the cells $i+\texttt{r}$, $i+\texttt{u}$, $i+\texttt{l}$ and $i+\texttt{d}$ at the same time step, where $\texttt{r}=(1,0)$, $\texttt{u}=(0,1)$, $\texttt{l}=(-1,0)$ and $\texttt{d}=(0,-1)$.

Let $\mu : S \to \mathbb{Q}^{\geq 0}$ be a conserved energy. Without loss of generality, we can assume that $\mu(\diamond)=0$ for a state $\diamond \in S$ which we call *blank*. For every state $x \in S$ we define the *free* flows going out of x by looking at the configurations

$$\begin{array}{ccc} \diamond & \diamond & \diamond \\ \diamond & x & \diamond \\ \diamond & \diamond & \diamond \end{array} \quad \xrightarrow{F} \quad \begin{array}{cc} x_2 & x_3 \\ x_1 & x_4 \end{array} \tag{1.2}$$

That is, $\varphi_{\swarrow}(x) \triangleq \mu(f(\begin{smallmatrix} \diamond & & x \\ & \bullet & \\ \diamond & & \diamond \end{smallmatrix}))$, $\varphi_{\nwarrow}(x) \triangleq \mu(f(\begin{smallmatrix} \diamond & & \diamond \\ & \bullet & \\ \diamond & & x \end{smallmatrix}))$, $\varphi_{\nearrow}(x) \triangleq \mu(f(\begin{smallmatrix} \diamond & & \diamond \\ & \bullet & \\ x & & \diamond \end{smallmatrix}))$ and $\varphi_{\searrow}(x) \triangleq \mu(f(\begin{smallmatrix} x & & \diamond \\ & \bullet & \\ \diamond & & \diamond \end{smallmatrix}))$. By the conservation of μ, we have

$$\varphi_{\swarrow}(x)+\varphi_{\nwarrow}(x)+\varphi_{\nearrow}(x)+\varphi_{\searrow}(x)=\mu(x)\,. \tag{1.3}$$

When two states are put next to each other, their touching out-going flows *interfere* and as a result we have a flow *deflection* from one cell toward another.

$$\begin{array}{cccc} \diamond & \diamond & \diamond & \diamond \\ \diamond & x & y & \diamond \\ \diamond & \diamond & \diamond & \diamond \end{array} \quad \xrightarrow{F} \quad \begin{array}{ccc} x_2 & a & y_3 \\ x_1 & b & y_4 \end{array} \tag{1.4}$$

Specifically, for every $x, y \in S$, define

$$\psi_{\uparrow}(x \;\; y) \triangleq \max\left\{0,\, \mu(f(\begin{smallmatrix} \diamond & & \diamond \\ & \bullet & \\ x & & y \end{smallmatrix})) - \varphi_{\nearrow}(x) - \varphi_{\nwarrow}(y)\right\}\,, \tag{1.5}$$

and

$$\psi_{\downarrow}(x \;\; y) \triangleq \max\left\{0,\, \mu(f(\begin{smallmatrix} x & & y \\ & \bullet & \\ \diamond & & \diamond \end{smallmatrix})) - \varphi_{\searrow}(x) - \varphi_{\swarrow}(y)\right\}\,. \tag{1.6}$$

By the conservation of μ we have

$$\mu(f(\begin{smallmatrix} \diamond & & \diamond \\ & \bullet & \\ x & & y \end{smallmatrix})) \;=\; \varphi_{\nearrow}(x) + \varphi_{\nwarrow}(y) + \psi_{\uparrow}(x\ \ y) - \psi_{\downarrow}(x\ \ y)\,, \tag{1.7}$$

$$\mu(f(\begin{smallmatrix} x & & y \\ & \bullet & \\ \diamond & & \diamond \end{smallmatrix})) \;=\; \varphi_{\searrow}(x) + \varphi_{\swarrow}(y) + \psi_{\downarrow}(x\ \ y) - \psi_{\uparrow}(x\ \ y)\,, \tag{1.8}$$

and either $\psi_{\downarrow}(x\ \ y)$ or $\psi_{\uparrow}(x\ \ y)$ is zero. The deflections $\psi_{\rightarrow}\binom{x}{y}$ and $\psi_{\leftarrow}\binom{x}{y}$ are defined similarly, and in the same way we have

$$\mu(f(\begin{smallmatrix} x & & \diamond \\ & \bullet & \\ y & & \diamond \end{smallmatrix})) \;=\; \varphi_{\searrow}(x) + \varphi_{\nearrow}(y) + \psi_{\rightarrow}\tbinom{x}{y} - \psi_{\leftarrow}\tbinom{x}{y}\,, \tag{1.9}$$

$$\mu(f(\begin{smallmatrix} \diamond & & x \\ & \bullet & \\ \diamond & & y \end{smallmatrix})) \;=\; \varphi_{\swarrow}(x) + \varphi_{\nwarrow}(y) + \psi_{\leftarrow}\tbinom{x}{y} - \psi_{\rightarrow}\tbinom{x}{y}\,, \tag{1.10}$$

and either $\psi_{\leftarrow}\binom{x}{y}$ or $\psi_{\rightarrow}\binom{x}{y}$ is zero. The deflections summarize all the interactions between the free flows:

Lemma 1.1. *For every $x, y, z, t \in S$ we have*

$$\begin{aligned} \mu(f(\begin{smallmatrix} y & & z \\ & \bullet & \\ x & & t \end{smallmatrix})) \;=\;& \varphi_{\nearrow}(x) + \varphi_{\searrow}(y) + \varphi_{\swarrow}(z) + \varphi_{\nwarrow}(t) \\ +\;& \psi_{\rightarrow}\tbinom{y}{x} + \psi_{\downarrow}(y\ \ z) + \psi_{\leftarrow}\tbinom{z}{t} + \psi_{\uparrow}(x\ \ t) \\ -\;& \psi_{\leftarrow}\tbinom{y}{x} - \psi_{\uparrow}(y\ \ z) - \psi_{\rightarrow}\tbinom{z}{t} - \psi_{\downarrow}(x\ \ t)\,. \end{aligned} \tag{1.11}$$

We shall think of the free flows and the flow deflections as weighted arrows from one cell (in the space-time) to another. For example, in the consecutive configurations

$$\begin{tikzcd} p \arrow[dr] & & y \arrow[dl] \arrow[dr] & & z \arrow[dl] & & & p & & y & & z \\ & a & & b & & \arrow[r, "F"] & & & a \arrow[rr] & & b & \\ q \arrow[ur] & & x \arrow[ul] \arrow[ur] & & t \arrow[ul] & & & q & & x & & t \end{tikzcd} \tag{1.12}$$

the free flow $\varphi_{\searrow}(p)$ determines an arrow toward the cell with state a from its up-left neighbor in the previous time step, and so forth. Similarly, there is a deflection arrow from a to b with weight $\psi_{\rightarrow}\binom{y}{x} \geq 0$ and one in the opposite direction with weight $\psi_{\leftarrow}\binom{y}{x} \geq 0$, even though at least one of them is zero. For two consecutive configurations x and $y = F(x)$, let us write $\Phi_{\nearrow}[i]$ for the free flow arrow with value $\varphi_{\nearrow}(x[i+\mathtt{dl}])$ from the cell $i+\mathtt{dl}$ (on x), to the cell i (on y), and so forth. Similarly, let $\Psi_{\uparrow}[i]$, $\Psi_{\rightarrow}[i]$, $\Psi_{\downarrow}[i]$ and $\Psi_{\leftarrow}[i]$ be the deflection arrows going out from i (on y) to the cells $i+\mathtt{u}$, $i+\mathtt{r}$, $i+\mathtt{d}$ and $i+\mathtt{l}$ (on y), respectively.

Deflections represent the deviation of an actual flow from the free flows. If we split each deflection $\psi_{\uparrow}$ (resp. $\psi_{\rightarrow}$, $\psi_{\downarrow}$, $\psi_{\leftarrow}$) into two parts $\psi_{\upharpoonleft}$ and $\psi_{\upharpoonright}$ (resp. $\psi_{\rightharpoonup}$ and $\psi_{\rightharpoondown}$, $\psi_{\downharpoonright}$ and $\psi_{\downharpoonleft}$, $\psi_{\leftharpoondown}$ and $\psi_{\leftharpoonup}$) and use these parts to correct the free flows, we obtain an actual flow for μ. To be precise, at each cell i, the arrow $\Phi_{\nearrow}[i]$ is corrected to

$$\Phi'_{\nearrow}[i] \triangleq \Phi_{\nearrow}[i] - \Psi_{\upharpoonright}[i] + \Psi_{\downharpoonright}[i+\mathtt{u}] - \Psi_{\rightharpoonup}[i] + \Psi_{\leftharpoonup}[i+\mathtt{r}] \tag{1.13}$$

and so forth (Figure 1). We require the splits $\psi_{\upharpoonleft}, \psi_{\upharpoonright}, \ldots$ to be non-negative and rational. Otherwise, the splitting can be arbitrary and may depend on the local neighborhood pattern. The main challenge here is to do the splitting in such a way that the corrected flow has only non-negative rational values.

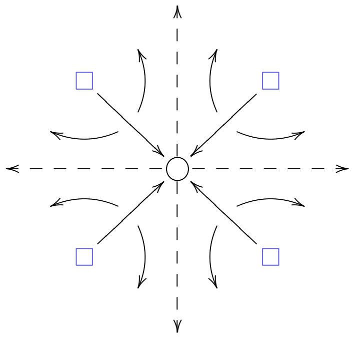


Figure 1: Correcting the flows.

Let us say that a cell i on y is *balanced*, if

a) $\Phi_{\searrow}[i] + \Phi_{\swarrow}[i] \geq \Psi_{\uparrow}[i]$ (and its rotations),

b) $\Phi_{\searrow}[i] + \Phi_{\swarrow}[i] + \Phi_{\nwarrow}[i] \geq \Psi_{\uparrow}[i] + \Psi_{\rightarrow}[i]$ (and its rotations), and

c) $\Phi_{\searrow}[i] + \Phi_{\swarrow}[i] + \Phi_{\nwarrow}[i] + \Phi_{\nearrow}[i] \geq \Psi_{\uparrow}[i] + \Psi_{\rightarrow}[i] + \Psi_{\downarrow}[i] + \Psi_{\leftarrow}[i]$.

Observation 1.2. For every $a, b, c \in S$ we have

a) $\varphi_{\searrow}(a) + \varphi_{\swarrow}(b) \geq \psi_{\uparrow}(\begin{smallmatrix} a & b \end{smallmatrix})$.

b) $\varphi_{\searrow}(a) + \varphi_{\swarrow}(b) + \varphi_{\nwarrow}(c) \geq \psi_{\uparrow}(\begin{smallmatrix} a & b \end{smallmatrix}) + \psi_{\rightarrow}\binom{b}{c}$.

Lemma 1.3. *If a cell is balanced, we can split its out-going deflections properly, so that its corrected in-coming flows remain non-negative and rational.*

Proof. Let i be a balanced cell. Let us do the splitting in such a way that (the splits are non-negative and rational, and) when the flows are corrected, the total amount of negative flows coming into i (let us call it M) is minimal. We claim that the corrected in-coming flows of i are non-negative.

Suppose the contrary. Without loss of generality, assume that $\Phi'_{\nearrow} < 0$. Since the sum of the corrected in-coming flows of i is non-negative (requirement (c) of balancedness), at least one of the other in-coming flows is strictly positive.

First assume that all the splits are strictly positive. If $\Phi'_{\nwarrow} > 0$, we could get a splitting with smaller M, by choosing

$$\Psi'_{\downharpoonleft} = \Psi_{\downharpoonleft} - \varepsilon \tag{1.14}$$

$$\Psi'_{\downharpoonright} = \Psi_{\downharpoonright} + \varepsilon \tag{1.15}$$

(see Figure 2.a) for a sufficiently small $\varepsilon > 0$.

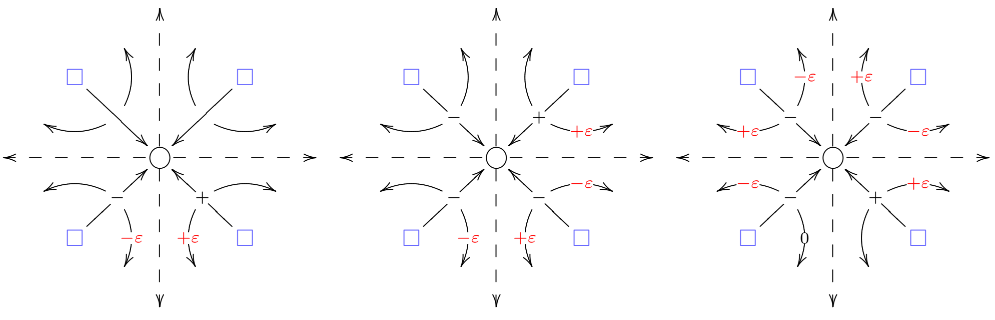


Figure 2: Proper splitting of the deflections.

By symmetry, $\Phi'_{\searrow} > 0$ cannot happen either. So let $\Phi'_{\nwarrow}, \Phi'_{\searrow} \leq 0$ and $\Phi'_{\swarrow} > 0$. Again, this is not possible, because if (for a sufficiently small $\varepsilon > 0$) we chose

$$\Psi'_{\downharpoonleft} = \Psi_{\downharpoonleft} - \varepsilon \tag{1.16}$$

$$\Psi'_{\downharpoonright} = \Psi_{\downharpoonright} + \varepsilon \tag{1.17}$$

$$\Psi'_{\rightharpoondown} = \Psi_{\rightharpoondown} - \varepsilon \tag{1.18}$$

$$\Psi'_{\rightharpoonup} = \Psi_{\rightharpoonup} + \varepsilon \tag{1.19}$$

(see Figure 2.b) we would get a smaller M.

If $\Psi_{\downharpoonleft} = 0$, then $\Phi_{\nearrow} < \Psi_{\leftharpoondown}$ and by requirement (a) of balancedness $\Phi_{\searrow} > \Psi_{\leftharpoonup} \geq 0$. If $\Phi'_{\searrow} > 0$, by choosing a different splitting like before, we could get a smaller M. So $\Phi'_{\searrow} \leq 0$ and $\Psi_{\upharpoonleft} > 0$. Now, by requirement (b) of balancedness $\Phi_{\swarrow} > \Psi_{\upharpoonright} \geq 0$. Again, if $\Phi'_{\swarrow} > 0$, by choosing a different splitting like before, leads us to a smaller M. So $\Phi'_{\swarrow} \leq 0$ and $\Psi_{\rightharpoonup} > 0$.

At this point, since $\Phi'_{\nearrow} < 0$ and $\Phi'_{\searrow}, \Phi'_{\swarrow} \leq 0$, we must have $\Phi'_{\nwarrow} > 0$. But again, by taking a sufficiently small $\varepsilon > 0$ and choosing the splitting

$$\Psi'_{\leftharpoondown} = \Psi_{\leftharpoondown} - \varepsilon \tag{1.20}$$

$$\Psi'_{\leftharpoonup} = \Psi_{\leftharpoonup} + \varepsilon \tag{1.21}$$

$$\Psi'_{\upharpoonleft} = \Psi_{\upharpoonleft} - \varepsilon \tag{1.22}$$

$$\Psi'_{\upharpoonright} = \Psi_{\upharpoonright} + \varepsilon \tag{1.23}$$

$$\Psi'_{\rightharpoonup} = \Psi_{\rightharpoonup} - \varepsilon \tag{1.24}$$

$$\Psi'_{\rightharpoondown} = \Psi_{\rightharpoondown} + \varepsilon \tag{1.25}$$

(see Figure 2.c) we would get a smaller M.

Finally, let $\Psi_{\downharpoonleft}, \Psi_{\leftharpoondown} > 0$. If either $\Phi'_{\nwarrow} > 0$ or $\Phi'_{\searrow} > 0$ we could make M smaller like before. So $\Phi'_{\nwarrow}, \Phi'_{\searrow} \leq 0$ and $\Phi'_{\swarrow} > 0$. Now, by the requirement (b) of balancedness, either $\Psi_{\rightharpoondown} > 0$ or $\Psi_{\upharpoonleft} > 0$. So, again we could get a smaller M as before.

Hence, in any case $\Phi'_{\nearrow} < 0$ leads to a contradiction, which by symmetry means the corrected in-coming flows of i must all be non-negative. ■

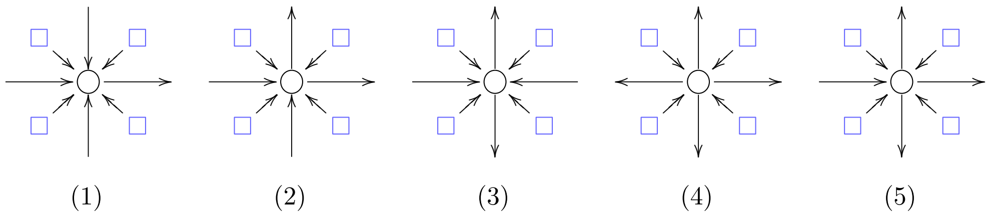


Figure 3: The situations that may happen around a cell.

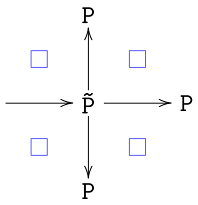


Figure 4: Doubly problematic cell.

Figure 3 shows a cell and the various situations that may happen, based on the direction of the deflection arrows around it. The other possibility are all symmetrically identical to these five case. According to the Lemma 1.3, Observation 1.2 guarantees that, unless there is exactly one deflection directing toward a cell (i.e., the cases (1-4)), one can correct the in-coming free flows of that cell to satisfy its out-going deflections, in such a way that the corrected flows remain non-negative. Let us call a cell *problematic* ($\mathtt{P}$ in symbol) if the situation around it is as in case (5) (or its symmetrically identical variants). We call a cell *doubly problematic* ($\tilde{\mathtt{P}}$ in symbol) if it is problematic, and furthermore, the endpoints of its out-going deflection arrows are also problematic (Figure 4). For two adjacent cells i and j, let us say j *follows* i, if there is a deflection arrow from i to j.

Observation 1.4. If j follows i, both of i and j cannot be doubly problematic at the same time.

Theorem 1.5. *Let $F : S^{\mathbb{Z}^2} \to S^{\mathbb{Z}^2}$ be a two-dimensional radius-$\frac{1}{2}$ cellular automaton. Then, every conserved energy $\mu : S \to \mathbb{Q}^{\geq 0}$ has a particle flow, with flows entering each cell only from its four neighbors.*

Proof. Let x and y be two consecutive configurations in $S^{\mathbb{Z}^2}$. Let us set the free flows as an initial approximation of the desired flow. That is, let $\Phi_d^{(0)} \triangleq \Phi_d$ ($d \in \{\nearrow, \searrow, \swarrow, \nwarrow\}$). We construct a non-negative rational flow for μ, by correcting this approximation in three steps. At each step a number of deflections are split and redistributed into the affected flows.

Step 1.
SPLITTING. For every cell i which is in either cases (1-4) of Figure 3 we split the out-going deflections as suggested in Lemma 1.3. If the cell is doubly problematic, we leave the splitting of its out-going deflections for the next step. If the cell is problematic, but not

doubly problematic, we leave one of its out-going deflections that leads to a non-problematic cell for the next step, and split the other two, as described in Lemma 1.3.
Correcting. We use the already split deflections to correct the flows. Let $\Phi_d^{(1)}$ ($d \in \{\nearrow, \searrow, \swarrow, \nwarrow\}$) be the corrected flow arrows of this step.

Step 2.
Splitting. Let i be a problematic cell. Notice that unless i follows a doubly problematic cell, its in-coming deflection is already resolved in the previous step. So, in this case i is no more problematic, and we can split its out-going deflections as explained in Lemma 1.3. In particular, all the out-going deflections of (formerly) doubly problematic cells are split in this step (Observation 1.4).
Correcting. We correct the flows using the newly split deflections. Let $\Phi_d^{(2)}$ ($d \in \{\nearrow, \searrow, \swarrow, \nwarrow\}$) be the corrected flow arrows of this step.

Step 3.
Splitting. The only unresolved deflections are those leaving a problematic cell (such as i) which follows an initially doubly problematic cell. But the out-going deflections of the doubly problematic cells are already resolved. So i is no further problematic. We split its unresolved out-going deflection using Lemma 1.3.
Correcting. We correct the flows using the newly split deflections. Let $\Phi_d^{(3)}$ ($d \in \{\nearrow, \searrow, \swarrow, \nwarrow\}$) be the corrected flow arrows of this step.

At this point, all the deflections are resolved. The corrected arrows $\Phi_d^{(3)}$ define a flow Φ by

$$\Phi_{i\to j} \triangleq \begin{cases} \Phi_{\nearrow}^{(3)}[j] & \text{if } i = j + \texttt{dl}, \\ \Phi_{\searrow}^{(3)}[j] & \text{if } i = j + \texttt{ul}, \\ \Phi_{\swarrow}^{(3)}[j] & \text{if } i = j + \texttt{ur}, \\ \Phi_{\nwarrow}^{(3)}[j] & \text{if } i = j + \texttt{dr}, \\ 0 & \text{otherwise}, \end{cases} \tag{1.26}$$

for μ, which satisfies the continuity equations, and its values are locally determined. Also, by construction, the values of Φ are all non-negative and rational. Therefore, Φ is a particle flow. ∎

2. Particle Flows in CA with Arbitrary Neighborhood

Every CA can be transformed into a radius-$\frac{1}{2}$ one, using a combination of a translation and moving to a higher block representation. A conserved energy $\mu : S \to \mathbb{Q}^{\geq 0}$ gives a conserved energy $\hat{\mu} : \hat{S} \to \mathbb{Q}^{\geq 0}$ for the new CA, which simply measures the collective energy of the super-cells of this new CA. By the discussion of the previous chapter, we can find a particle flow for $\hat{\mu}$. This can be turned into a particle flow for μ in the following way.

Let x and $y = F(x)$ be two consecutive configurations in the original CA. For each super-cell $\hat{i}$ on x let us order its out-going flow arrows using, for example, the lexicographic ordering of their end points, and distribute them properly into its constituent cells, so that each cell with state s gives out exactly $\mu(s)$. Now for each cell i on x and each super-cell $\hat{j}$ on y, we have a flow arrow from i to $\hat{j}$. Then, for each super-cell on y order its in-coming flow arrows according to the lexicographic ordering of their starting points, and distribute them properly into its constituent cells, so that each cell with state s receives exactly $\mu(s)$. Clearly, this can be done locally, but the obtained flow may not be translation-invariant. However, if we translate the partitioning of the cells and take the average of the flows obtained from each partitioning, we obtain a translation-invariant flow which is still non-negative and rational-valued.

3. Open Problems

With enough patience, one should be able to find a particle representation for the conservation laws of the three-dimensional CA, or the CA on the hexagonal or triangular lattices, using similar analysis. Is there a unified approach that works for any lattice, in any number of dimensions?

A drawback of our solution is the arbitrariness involved. There are infinite number of ways one can assign a flow to a given conservation law. Can we (possibly by putting some extra constraints, or by formalizing the concept of the flows in a different way) obtain a "natural" flow for each conservation law, which is *unique*? One criterion for naturalness is that for a reversible CA, the flows in the backward direction of time should be obtained from the flows in the forward direction, only by reversing the direction of the arrows. Such a concept of flow would definitely give a better understanding of the dynamics of the conserved energy.

References

[1] N. Boccara and H. Fukś, *Motion representation of one-dimensional cellular automata rules*, International Journal of Modern Physics C, 17 (2006), pp. 1605–1611.

[2] T. Boykett, J. Kari, and S. Taati, *Conservation laws in rectangular CA*, Journal of Cellular Automata, 3 (2008), pp. 115–122.

[3] T. Boykett and C. Moore, *Conserved quantities in one-dimensional cellular automata.* Unpublished manuscript, 1998.

[4] B. Durand, E. Formenti, and Z. Róka, *Number conserving cellular automata I: decidability*, Theoretical Computer Science, 299 (2003), pp. 523–535.

[5] E. Formenti and A. Grange, *Number conserving cellular automata II: dynamics*, Theoretical Computer Science, 304 (2003), pp. 269–290.

[6] H. Fukś, *A class of cellular automata equivalent to deterministic particle systems*, in Hydrodynamic Limits and Related Topics, S. Feng, A. T. Lawniczak, and S. R. S. Varadhan, eds., vol. 27 of Fields Institute Communications, American Mathematical Society, 2000, pp. 57–69.

[7] T. Hattori and S. Takesue, *Additive conserved quantities in discrete-time lattice dynamical systems*, Physica D, 49 (1991), pp. 295–322.

[8] J. Kari, *Theory of cellular automata: A survey*, Theoretical Computer Science, 334 (2005), pp. 3–33.

[9] A. Moreira, N. Boccara, and E. Goles, *On conservative and monotone one-dimensional cellular automata and their particle representation*, Theoretical Computer Science, 325 (2004), pp. 285–316.

[10] M. Pivato, *Conservation laws in cellular automata*, Nonlinearity, 15 (2002), pp. 1781–1793.

Journées Automates Cellulaires 2008 (Uzès), pp. 74-89

GÖDEL INCOMPLETENESS REVISITED

GREGORY LAFITTE [1]

[1] Laboratoire d'Informatique Fondamentale de Marseille (LIF), CNRS – Aix-Marseille Université, 39 rue Joliot-Curie, 13453 Marseille Cedex 13, France
E-mail address: `Gregory.Lafitte@lif.univ-mrs.fr`
URL: `http://www.lif.univ-mrs.fr/~lafitte/`

ABSTRACT. We investigate the frontline of Gödel's incompleteness theorems' proofs and the links with computability.

The Gödel incompleteness phenomenon

Gödel's incompleteness theorems [Göd31, SFKM$^+$86] are milestones in the subject of mathematical logic.

Apart from Gödel's original syntactical proof, many other proofs have been presented. Kreisel's proof [Kre68] was the first with a model-theoretical flavor. Most of these proofs are attempts to get rid of any form of self-referential reasoning, even if there remains diagonalization arguments in each of these proofs. The reason for this quest holds in the fact that the diagonalization lemma, when used as a method of constructing an independent statement, is intuitively unclear. Boolos' proof [Boo89b] was the first attempt in this direction and gave rise to many other attempts. Sometimes, it unfortunately sounds a bit like finding a way to sweep self-reference under the mathematical rug.

One of these attempts has been to prove the incompleteness theorems using another paradox than the Richard and the Liar paradoxes. It is interesting to note that, in his famous paper announcing the incompleteness theorem, Gödel remarked that, though his argument is analogous to the Liar paradox, "Any epistemological antinomy could be used for a similar proof of the existence of undecidable propositions". G. Boolos has proved quite recently (1989) a form of the first incompleteness theorem using Berry's paradox consisting in the fact that "the least integer not nameable in fewer than seventy characters" has just now been named in sixty-three characters. G. Boolos thought the interest of such proofs is that they provide a *different sort of reason* for incompleteness. It is true that each of these new arguments gives us a better understanding of the incompleteness phenomenon.

When studying proofs and provability, there are two different points of view: the proof-theoretical one (axioms and inference rules) and the model-theoretical one (axioms, models, consequences). The former one tends to be quite syntactical and the latter one more semantical. We have tried to present both points of view and linger over the model-theoretical side because, at least from the author's point of view, model-theoretic arguments are intuitively clearer than proof-theoretic ones.

Gödel's argumentation was heavily based not only on the arithmetization of syntax, but on the arithmetization of all mathematical objects (sentences, proofs, theories) and the fact that all this arithmetization is primitive recursive. In fact, it has opened the way for the notions of computation and computability to arise.

The goal of this paper is twofold: a survey of incompleteness proofs and to precise links with computability.

Computability and incompleteness are inherently linked. For instance, one can obtain a first form of the first incompleteness theorem by considering propositions of the form $n \notin X$, where X is a non-recursive but recursively enumerable set, *e.g.*, the *diagonal halting* set $\mathcal{K}$. Even if the language of the considered theory does not contain $\in$, there is a simple algorithm that generates given n the proposition "$n \notin X$". Given a sound (every provable statement is true) recursively enumerable theory T, there is a number n_0 such that $n_0 \notin X$ but T does not prove it. The proof is direct: Suppose that there is no such n_0, then we would have that T proves "$n \notin X$" if and only $n \notin X$, and X would be recursive (generate the theorems of T and at the same time enumerate X; if $n \in X$ then n will eventually show up in the enumeration; otherwise, "$n \notin X$" will eventually show up in the theorems of T and be true by the soundness assumption). We thus have a true sentence, "$n_0 \notin X$", which is not provable in T.

Incompleteness is also famously linked to computability via Chaitin's incompleteness theorem. Chaitin's result, showing that there are unprovable statements on Kolmogorov-Chaitin complexity[1], is a form of Gödel's first incompleteness theorem. Actually, Kolmogorov showed in the sixties that the set of non-random (or incompressible) numbers, *i.e.*, $\{x : K(x) \geqslant x\}$, is recursively enumerable but not recursive, and, by the above argument, this is already a version of Gödel's first incompleteness theorem. Moreover, Kolmogorov's proof can be seen as an application of Berry's paradox. Following Boolos, it is thus no wonder that we can get proofs using this Kolmogorov complexity function (or other *similar* computability-related functions) of both incompleteness theorems.

One of the reason of the existence of the quest of better understanding the incompleteness phenomenon holds in the peculiarity of Gödel's unprovable statements. They are not natural mathematical statements: no mathematician has ever stumbled on them (or should we say *over them*?). And thus, it seems to many that normal mathematical practice is not concerned with the incompleteness phenomenon. More and more results show however the contrary. In particular, Harvey Friedman's Π^0_1 statements, that are unprovable in Zermelo-Fraenkel (ZF) set theory and need the 1-consistency of strong set-theoretical unprovable statements, going way beyond ZF, to be proved, are examples of such results.

Nevertheless, incompleteness theorems only provide unprovable statements like the *consistency* of a theory, that are of an unclear nature. What combinatorial properties does the consistency statement bring to a theory? Feferman [Fef62, Her88] has shown that a certain reflection principle, an unprovable statement, has to be *added* $\omega^{\omega^{\omega+1}}$ times to Peano arithmetic in order to *cover* all true arithmetical statements. Adding the 1-consistency, a soundness assumption, of strong set-theoretical unprovable statements, *e.g.,* large cardinal axioms, to a given arithmetical theory amounts to asserting that "every Π^0_2 consequence of these strong statements is true". In this case, the combinatorial properties that are *added* are the combinatorial Π^0_2 consequences of these strong statements. Having a link between consistency (or soundness) and computability, in particular Kolmogorov complexity, would

[1]Loosely speaking, the Kolmogorov-Chaitin complexity of a natural number n, denoted by $K(n)$, is the smallest size of a program which generates n.

make possible an understanding of what properties consistency adds to a theory. Adding consistency as an axiom would then yield new combinatorial properties because of the existing links between combinatorics and Kolmogorov complexity. This could be one reason behind the stir surrounding Chaitin's incompleteness result.

This paper is organized as follows. We start by recalling the basic notions behind formulæ, proofs, theories and arithmetization. Then we present Gödel's original proofs. We continue by presenting a survey of existing incompleteness proofs, of both first and second incompleteness theorems. We finish with incompleteness results and proofs that are computability-related and discuss the interpretation of Chaitin's incompleteness theorem.

1. Objects to play with

1.1. What are the basic objects?

On top of the usual logical connectives ($\wedge$, $\vee$ and $\neg$), we will respectively denote the logical connectives of *implication* and *equivalence* by $\rhd$ and $\equiv$.

$\mathcal{L}_{\mathrm{PA}}$ will designate the first order language on the signature of arithmetic $\{\mathbf{S}, +, \times, \leqslant, \mathbf{0}\}$. $\mathbf{S}$, $+$, $\times$ designate respectively the *successor*, *addition* and *multiplication* functions. $\leqslant$ designates the *lower-or-equal* relation and $\mathbf{0}$ designates the constant *zero*.

The Turing machines indexed by their codes, for any appropriate coding which we will later on make to coincide with the Gödel numbering, are denoted by $\{T_i\}_{i\in\mathbb{N}}$. A computation (of a Turing machine, or any equivalent computation model) either *diverges*, denoted by $\uparrow$, or *converges*, denoted by $\downarrow$. The partial recursive functions computed by Turing machines, following a fixed convention, are denoted by $\{\varphi_i\}_{i\in\mathbb{N}}$ (agreeing with the Turing machines' coding). The sets $\{W_i\}_{i\in\mathbb{N}}$ denote the recursively enumerable sets, *i.e.*, the domains of partial recursive functions. A central set in computability theory is the *diagonal halting set* $\mathcal{K} = \{x : \varphi_x(x)\downarrow\} = \{x : T_x(x)\downarrow\} = \{x : x \in W_x\}$. The set $\mathcal{K}$ is recursively enumerable but not recursive; it is the archetypal *creative* set.

Concerning computability, the reader is referred to [Odi89, Odi99, Rog67, Rog58, Smu93, VS03].

1.2. What is a proof?

A formal theory T is determined by a first order[2] language $\mathcal{L}_T$ and a set of axioms $\mathcal{A}_T$, which are formulæ in that language. The set Thms_T of *theorems* consists of those formulæ ϕ for which there is a proof in T.

There are two different points of view concerning proofs: the proof-theoretical one (axioms and inference rules) and the model-theoretical one (axioms, models, consequences).

[2]All our reasoning also works for theories on second order languages. For simplicity and brevity, we will only consider first order theories in this article.

1.2.1. *Proof theoretical.* Proofs are most commonly seen as a deduction sequence from a set of axioms.

In proof theory, we can for example take the following deduction rules:

$\Gamma \Rightarrow \phi$ if $\phi \in \Gamma$	$\Gamma \Rightarrow \phi \wedge \psi$ iff $\Gamma \Rightarrow$ and $\Gamma \Rightarrow \psi$
If $\Gamma \Rightarrow \phi$ or $\Gamma \Rightarrow \psi$, then $\Gamma \Rightarrow \phi \vee \psi$	If $\Gamma \cup \{\phi\} \Rightarrow \psi$, then $\Gamma \Rightarrow \phi \rhd \psi$
If $\Gamma \Rightarrow \phi$ and $\Gamma \Rightarrow \phi \rhd \psi$, then $\Gamma \Rightarrow \psi$	If $\Gamma \Rightarrow (s = t)$ and $\Gamma \Rightarrow \phi(s)_x$, then $\Gamma \Rightarrow \phi(t)_x$
If $\Gamma \Rightarrow \psi$ and $\Gamma \Rightarrow \neg\psi$, then $\Gamma \Rightarrow \phi$	$\Gamma \Rightarrow \forall x\ (x = x)$
If $\Gamma \cup \{\neg\phi\} \Rightarrow \psi$ and $\Gamma \cup \{\neg\phi\} \Rightarrow \neg\psi$, then $\Gamma \Rightarrow \phi$	
If $\Gamma \cup \{\phi\} \Rightarrow \psi$ and $\Gamma \cup \{\phi\} \Rightarrow \neg\psi$, then $\Gamma \Rightarrow \neg\phi$	
If $\Gamma \cup \{\phi\} \Rightarrow \theta$ and $\Gamma \cup \{\psi\} \Rightarrow \theta$, then $\Gamma \cup \{\phi \vee \psi\} \Rightarrow \theta$	
If $\Gamma \Rightarrow \phi$ and x does not occur free in Γ, then $\Gamma \cup \Delta \Rightarrow \forall x\ \phi$	
If $\Gamma \Rightarrow \forall x\ \phi$, then $\Gamma \Rightarrow \phi(s)_x$ for any term s free for x in ϕ	
If $\Gamma \Rightarrow \phi(s)_x$, then $\Gamma \Rightarrow \exists x\ \phi$, for any term s free for x in ϕ	
If $\Gamma \cup \{\phi(y)_x\} \Rightarrow \psi$ and y is not free in Γ or ψ, then $\Gamma \cup \Delta \cup \{\exists x\ \phi\} \Rightarrow \psi$	
If $\Gamma \Rightarrow \forall x\ (x \in X \equiv x \in Y)$, then $\Gamma \Rightarrow X = Y$	

$\Delta \Rightarrow \phi$ holds if and only if there is a derivation showing this in the form of a finite sequence $\langle \Gamma_1, \phi_1 \rangle$, $\ldots$, $\langle \Gamma_n, \phi_n \rangle$, where $\langle \Gamma_n, \phi_n \rangle$ is $\langle \Delta, \phi \rangle$ and each $\langle \Gamma_i, \phi_i \rangle$ follows by one of the above rules from previous pairs in the sequence. A *derivation* is a sequence number $\langle s_0, \ldots, s_n \rangle$ where each s_i is a pair $\langle t_i, \phi_i \rangle$ with t_i a sequence number of formulæ and s_i is related as indicated in the rules to zero, one or two previous pairs in the sequence. s is a derivation *of* ϕ *from* Γ if s_n is $\langle t_n, \phi \rangle$ where every formula in the sequence t_n is a member of Γ. A *proof* in a theory T is a derivation from $\mathcal{A}_T$.

1.2.2. *Model theoretical.* Another way to consider provability is through models. A sentence ϕ is provable in an axiomatic theory T if all models of T satisfy ϕ.

Leon Henkin gave in 1949 a non-constructive but easier (than Gödel's original) proof of Gödel's completeness theorem. It consists in reducing the consistency of a set of sentences in a language L to that of a set of quantifier-free sentences in an extended language. This process can be arithmetized to build a partial order, called the *Henkin tree*. It gives an arithmetical Δ_{n+1} model for any consistent Σ_n or Δ_n theory. For a complete description, the reader is referred to [Kay91].

Through Henkin's method, we can obtain a more model-theoretical notion of proof. If ϕ and ψ are sentences, to say that ψ is a consequence of ϕ is to say that the set $\{\phi, \neg\psi\}$ is inconsistent. The consistency of $\{\phi, \neg\psi\}$ can be determined by Henkin's method. We get a proof that ψ is a consequence of ϕ as soon as we reach a natural number p at which the branches of Henkin's tree all end at a contradiction. This natural number p can take the place of a proof.

If a theory T is a Σ_n fragment of arithmetic, then consistency can be expressed by a Π_n sentence: it is enough to express that $\mathbf{0} = \mathbf{S0}$ is not a consequence of the axioms of T or else to express that the Henkin tree associated with T is infinite.

For more on model theory, see [Hod93].

1.3. What is an arithmetical-able theory?

Throughout this paper, T will be some fixed, but unspecified, consistent formal theory.

The properties that a theory T should meet to satisfy the clauses of incompleteness theorems are for it to *contain arithmetic.* There are several ways to precise these properties. These properties are encodability conditions and, as Gödel showed, one can do a great deal of encoding on natural numbers.

To follow classical expositions of the incompleteness theorems, we assume that the encoding is done in some fixed formal theory S and that T contains S. S is usually not specified but it is commonly taken to be a formal system of arithmetic, although a weak set theory[3] is often more convenient. If S is a formal system of arithmetic, *e.g.*, PA (Peano Arithmetic), and T is ZF (Zermelo-Fraenkel set theory), then T contains S in the sense that there is a well-known embedding of S in T.

S needs to be able to represent primitive recursive functions. It should be *primitive recursive*-able. A more model-theoretical way to require these properties is to require of the theory to have Σ_1-induction.

To each formula ϕ of the language of T is assigned a closed term, $\ulcorner\phi\urcorner$, called the *code* of ϕ. For any natural number n, $\ulcorner n\urcorner$ designates a closed term, the *numeral* for n, in the language S that represents n, *e.g.*, $\sigma(\sigma(\ldots(0)\ldots))$. n is called the *value* of this numeral $\ulcorner n\urcorner$.

To avoid any ambiguity, we define a function called var. An ambiguity arises in the following example. There are two possible meanings for $\ulcorner x\urcorner$: the code for the value for the variable x *or* the code for the variable x. $\mathrm{var}(x)$ designates the latter case, *i.e.*, the code for the variable x.

S will have certain function symbols corresponding to the logical connectives and quantifiers : neg, implies, *etc.*, such that, for all formulæ ϕ, ψ, $S \vdash \mathrm{neg}(\ulcorner\phi\urcorner) = \ulcorner\neg\phi\urcorner$, $S \vdash \mathrm{implies}(\ulcorner\phi\urcorner, \ulcorner\psi\urcorner) = \ulcorner\phi \rhd \psi\urcorner$, *etc.*

The *substitution* operator, represented in S by the function symbol sub, is of particular importance. For any codes c_1 and c_2 for terms t_1 and t_2 and a variable x, $\mathrm{sub}_x(c_1, c_2)$ is the code of the term that results from substituting t_1 for every occurrence of x in t_2 : $S \vdash \mathrm{sub}_x(\ulcorner t\urcorner, \ulcorner\phi(x)\urcorner) = \ulcorner\phi(t)_x\urcorner$.

For readability, we will use the same names for functions and predicates in formulæ and in the running text. All the functions previously defined are actually primitive recursive.

From the previous discussion on proofs, we have a binary relation, whose symbol in S is Proof, such that for closed t_1 and t_2: $S \vdash \mathrm{Proof}_T(t_1, t_2)$ iff t_1 is the code of a *proof* in T of the formula with code t_2. It follows that $T \vdash \phi$ iff $S \vdash \mathrm{Proof}_T(t, \ulcorner\phi\urcorner)$ for some closed term t.

We then define a predicate, whose symbol in S is Prov, asserting provability :

$$\mathrm{Prov}_T(y) \equiv \exists x \ \mathrm{Proof}_T(x, y)$$

One must be careful and understand that we do not always have : $T \vdash \phi$ iff $S \vdash \mathrm{Prov}_T(\ulcorner\varphi\urcorner)$. It depends on the *soundness* properties of our theory. Soundness is linked to consistency as summarized in section 1.4.

We will use a special notation for formalizations of provability statements. If ϕ is a sentence, we write $\Box\phi$ for the sentence $\mathrm{Prov}_T(\ulcorner\phi\urcorner)$, where the theory T is implicit. Accordingly, $\Box\,\Box\,\phi$ is the formula $\mathrm{Prov}_T(\ulcorner\psi\urcorner)$ where ψ is $\mathrm{Prov}_T(\ulcorner\phi\urcorner)$.

[3]See [Dev84, Jec78].

This encoding (of S in T) can be carried out in such a way that the following important conditions, the *deducibility* (or *derivability*) *conditions*, are met for all sentences ϕ:

$$T \vdash \phi \text{ implies } S \vdash \mathrm{Prov}_T(\ulcorner \phi \urcorner), \text{ for every sentence } \phi. \tag{1.1}$$

$$S \vdash \mathrm{Prov}_T(\ulcorner \phi \urcorner) \rhd \mathrm{Prov}_T(\ulcorner \mathrm{Prov}_T(\ulcorner \phi \urcorner) \urcorner), \text{ for every sentence } \phi. \tag{1.2}$$

$$S \vdash \mathrm{Prov}_T(\ulcorner \phi \urcorner) \wedge \mathrm{Prov}_T(\ulcorner \phi \rhd \psi \urcorner) \rhd \mathrm{Prov}_T(\ulcorner \psi \urcorner), \text{ for all sentences } \phi, \psi. \tag{1.3}$$

Much of the intricacy of Gödel's incompleteness theorems' proofs lies in the scarcely illuminating details of setting up and checking the properties of a coding system representing the syntax of $\mathcal{L}_{\mathrm{PA}}$ within that same language. For this reason a number of efforts have been made to present the essentials of the proofs of Gödel's theorems without getting entangled in syntactic details. One of the most important of these efforts was made by Löb [Löb55] and Hilbert and Bernays [HB39]. They formulated these three conditions on the provability predicate in a formal system which are jointly sufficient to yield Gödel's second incompleteness theorem.

Given that the axioms of T are defined using a Σ-formula, these deducibility conditions all hold: the first two conditions are corollaries of the Σ-completeness theorem and the third condition is a formalization of an obvious argument.

1.4. What are *consistency* statements?

A theory T is *inconsistent* if there exists ϕ such that ϕ **and** $\neg\phi$ are theorems of T, and otherwise *consistent.*

A theory T is *complete* if for every sentence ϕ in the language of T, ϕ or $\neg\phi$ is a theorem of T, and otherwise *incomplete.*

Gödel introduced a stronger form of consistency, coined ω-consistency. In a w-consistent theory T, we cannot have at the same time $T \vdash \exists x\ \phi(x)$ and $T \vdash \neg\phi(\ulcorner 0 \urcorner), T \vdash \neg\phi(\ulcorner 1 \urcorner), \ldots$ (having for all natural number i, $T \vdash \neg\phi(\ulcorner i \urcorner)$).

More formally: T is *ω-consistent* if for any formula ϕ

$$\mathrm{Prov}_T(\ulcorner \exists x\ \phi(x) \urcorner) \text{ implies } \exists x\ \neg\, \mathrm{Prov}_T(\ulcorner \neg\phi(x) \urcorner) \tag{1.4}$$

ω-consistency is a restriction of another property, *reflection*:

$$\mathrm{Refl}_T: \quad \mathrm{Prov}_T(\ulcorner \phi \urcorner) \text{ implies } \phi \text{ for closed } \phi$$

For $\phi \in \Delta_0$, (1.4) is called 1-*consistency.* It can be shown that 1-consistency means that all Σ_1 provable statements are true. It is actually Refl_T for Σ_1 statements, denoted by $\mathrm{Refl}_T^{\Sigma_1}$. Reflection is also called *soundness.* Σ-soundness is 1-consistency

Nevertheless, every arithmetical-able theory T has the following property.

Theorem 1.1 (Σ_1-completeness). *If ϕ is a Σ_1 statement, then $S \vdash \phi \rhd \mathrm{Prov}_T(\ulcorner \phi \urcorner)$.*

Hence, in T, Cons_T, the statement expressing that there is no proof in T of $\mathbf{0} = \mathbf{S0}$, is equivalent to reflection of Π_1 statements, denoted by $\mathrm{Refl}_T^{\Pi_1}$.

Consistency statements play a major role in the incompleteness theorems. Each incompleteness result necessitates a consistency statement assumption on the considered theory. Plain consistency is the weakest of these statements. Gödel introduced ω-consistency to be able to obtain an independent statement. The weaker assumption, 1-consistency, generally suffices. In all our incompleteness theorems and proofs, the strongest assumption is 1-consistency. For more details on consistency and reflection statements, the reader is referred to [Smo77].

2. Original and model-theoretical proofs

For various descriptions of mathematical logic in general and Gödel's incompleteness theorems in particular, see [Smo77, Kle52, Fef60, Kre50, Kot94, Kot96, Kot98, Kot04, Ros36, Boo95, End72, Hen57].

2.1. Original (syntactical) proof

The original proof of Gödel's incompleteness theorems goes necessarily through proving the *diagonalization lemma.*

Lemma 2.1 (Diagonalization lemma)**.** *For every formula ψ with a single free variable x there is a sentence ϕ such that $S \vdash \phi \equiv \psi(\ulcorner\phi\urcorner)_x$.*

Proof. Given ψ, let θ_x be $\psi(\mathrm{sub}_x(\mathrm{var}(x), x))_x$, the diagonalization of ψ. Let $m = \ulcorner\theta_x\urcorner$ and $\phi = \theta(m)_x$. Then we have

$$S \vdash \phi \equiv \psi(\ulcorner\phi\urcorner)_x$$

In S, we have that

$$\phi \equiv \theta(m)_x \equiv \psi(\mathrm{sub}_x(\mathrm{var}(m), m))_x \equiv \psi(\mathrm{sub}_x(\mathrm{var}(m), \ulcorner\theta_x\urcorner))_x \equiv \psi(\ulcorner\theta(m)_x\urcorner)_x \equiv \psi(\ulcorner\phi\urcorner)_x$$

■

The[4] Gödel sentence G_T for T consists in diagonalizing $\neg\,\mathrm{Proof}_T(\cdot)$. By the diagonalization lemma, we have a sentence G_T such that $G_T \equiv \neg\,\square\,G_T$ is provable in S.

Theorem 2.2 (Gödel's first incompleteness theorem)**.** *If T is consistent, G_T is not provable in T, and if T is Σ-sound, then G_T is independent of T.*

Proof. If G_T is provable in T, then $\square G_T$ is also provable by the first deducibility condition. By definition of G_T, we thus have that $\neg G_T$ is provable in T, so T is inconsistent.

If $\neg G_T$ is provable in T, either T is inconsistent and thus not Σ-sound, or if T is consistent, $\neg G_T$ is false (since G_T is true: we have just proved that "G_T is not provable", which is equivalent in S to G_T) and again T is not Σ-sound, since $\neg G_T$ is equivalent in S (and thus in T) to $\square G_T$, which is equivalent in S to a Σ-formula. ■

[4]There is no uniquely defined Gödel sentence for a theory T, because the sentences depend on the Σ-formula used to define the axioms of T. It is an abuse of language.

This proof is formalizable in T: $\Box(G_T \rhd \neg \Box G_T)$ is provable in T by definition of G_T and 1.1, so $\Box G_T \rhd \Box\neg \Box G_T$ is provable in T by 1.3, and $\Box G_T \rhd \Box \Box G_T$ is provable in T by 1.2, so $\Box G_T \rhd (\Box \Box G_T \wedge \Box\neg \Box G_T)$ is provable in T. Hence, $\Box G_T \rhd \neg \mathrm{Cons}_T$ is provable in T.

Thus, Cons_T implies $\neg \Box G_T$ (and also G_T) in T.

This yields the second incompleteness theorem:

Theorem 2.3 (Gödel's second incompleteness theorem)**.** *If T is consistent,* Cons_T *is not provable in T.*

Proof. The formalization of theorem 2.2 shows that Cons_T implies G_T in T and by the same theorem 2.2, G_T cannot be provable in T, and thus in T neither can Cons_T.

The implication $G_T \rhd \mathrm{Cons}_T$ is also provable in T, since "if T is inconsistent, every formula is provable in T" is provable in T. Thus, G_T and Cons_T are in fact equivalent in T. ■

The second incompleteness theorem can also be strengthened:

Theorem 2.4 (Löb's theorem)**.** *If ϕ is a sentence for which $\Box\phi \rhd \phi$ is provable in T, then ϕ is provable in T.*

Kreisel's proof. If $\Box\phi \rhd \phi$ is provable in T, then $T + \neg\phi \vdash \neg \Box \phi$, which is equivalent in T to $\mathrm{Cons}_{T+\neg\phi}$. Thereby, $T + \neg\phi$ proves its own consistency, and so by theorem 2.3 is inconsistent. Thus ϕ is a theorem of T. ■

Löb's original proof. Suppose that $\Box\phi \rhd \phi$ is provable in T and let ψ be the diagonalization of $\Box x \rhd \phi$: the diagonal lemma gives ψ such that $\psi \equiv (\Box\psi \rhd \phi)$ is provable in T. Thus we have by 1.1 and two applications of 1.3 that

$$\Box\psi \rhd (\Box \Box \psi \rhd \Box\phi) \text{ is provable in } T.$$

By 1.2, we obtain that $\Box\psi \rhd \Box\phi$ is provable in T and also by the assumption on ϕ,

$$\Box\psi \rhd \phi \text{ is provable in } T. \tag{2.1}$$

By definition of ψ, it follows that ψ is provable in T. All of this has been proven in T, thus $\Box\psi$ is provable in T, and by 2.1, ϕ is provable in T. ■

Rosser's theorem is a variant of Gödel first incompleteness theorem dropping the soundness condition on T to obtain an independent statement. It uses a modification of Proof_T.

$$\mathrm{Proof}^R_T(x, \ulcorner y \urcorner) \text{ iff } \mathrm{Proof}_T(x, \ulcorner y \urcorner) \wedge \forall z, \ulcorner w \urcorner \leqslant x,\ (\mathrm{Prov}_T(z, \ulcorner w \urcorner) \rhd y \neq \neg w)$$

From Proof^R_T, one defines Prov^R_T and Cons^R_T.

Theorem 2.5 (Rosser's theorem)**.** *Let ϕ be a sentence by the diagonalization lemma such that $S \vdash \phi \equiv \neg \mathrm{Prov}^R_T(\ulcorner \phi \urcorner)$. Then*

(1) $T \nvdash \phi$;
(2) $T \nvdash \neg\phi$;
(3) $T \vdash \mathrm{Cons}^R_T$.

2.2. Semantical proofs

By semantical proofs, we mean "more model-theoretical" proofs.

2.2.1. $G_T \equiv \mathrm{Cons}_T$. From the first syntactical incompleteness theorem, one can obtain a model theoretical proof of the second incompleteness theorem: a model theoretical way to prove the equivalence of G_T and Cons_T. It is a forerunner of the ideas underlying the subsequent model-theoretical proofs of both incompleteness theorems.

Model-theoretical proof of $G_T \equiv \mathrm{Cons}_T$. Suppose that we have a model $\mathcal{M}$ of $T + \mathrm{Cons}_T$ such that $\mathcal{M} \models \neg G_T$.

Since $\mathcal{M} \models \mathrm{Cons}_T$, Henkin's completeness theorem gives a Δ_2 model $\mathcal{M}'$ such that $\mathcal{M}' \models T$.

$\mathcal{M} \models \neg G_T$, thus $\mathcal{M} \models \Box G_T$ and by Henkin's construction, $\mathcal{M}' \models G_T$.

In $\mathcal{M}$, we define a function which to $x \in \mathcal{M}$ gives $x_{\mathcal{M}'}$, the x-th successor of $\mathbf{0}$ in the sense of $\mathcal{M}'$. Let $\mathcal{M}''$ be the image of $\mathcal{M}$ by this function; it is an initial segment of $\mathcal{M}'$.

G_T is Π_1 and $\mathcal{M}' \models G_T$, thus so does $\mathcal{M}''$ and $\mathcal{M}$. *Contradiction* with $\mathcal{M} \models \neg G_T$.

■

2.2.2. *Chaitin's proofs of theorem 2.2.* Let $\{\varphi_i\}_{i\in\mathbb{N}}$ designate a recursive enumeration of all partial recursive functions. We work with an *acceptable* enumeration φ, in which all classical computability results hold (enumeration, *s-m-n*, fixed point, *etc.*).

For the purpose of proving Chaitin's incompleteness theorem, the following simple definition of *Kolmogorov complexity* is sufficient.

$$K_\varphi(x|y) = \text{ smallest } e \text{ such that } \varphi_e(y) = x, \text{ and } K_\varphi(x) = K_\varphi(x|0)$$

A more classical definition of Kolmogorov complexity goes as follows. A complexity is defined according to a *decompressor*, giving the length of a smallest input to the decompressor yielding the sought string. The Kolmogorov complexity is then the complexity according to an *optimal* decompressor; *optimal* in the sense that it differs only by an additive constant from other decompressors. For more on classical Kolmogorov complexity, see [LV90]. Our definition is merely a change of scale from the classical one. Both share the same basic computability properties: a computable function which is a lower bound for them is necessarily bounded and their graphs are Turing-complete. We call these functions the Kolmogorov functions.

Theorem 2.6 (Chaitin's theorem). *Let T be a arithmetical-able sound theory. There is a constant $\mathfrak{c}_T$ such that T does not prove "$K_\varphi(x) > \mathfrak{c}_T$" for any x.*

Proof. Let f be the recursive function assigning to c the code m of a the Turing machine M, such that M enumerates the theorems of T, searches for a theorem of the form "$K_\varphi(x) > c$" and in case of success, outputs x.

By Kleene's recursion theorem there is an e such that $\varphi_e = \varphi_{f(e)}$. Suppose that T_e halts when started with input 0. T_e outputs x such that $K_\varphi(x) > e$ because of the soundness assumption. On the other hand, if T_e outputs x with input 0, then $K_\varphi(x) \leqslant e$ by the definition of K_φ. *Contradiction.*

Hence, T_e does not halt and thus there is no proof of "$K_\varphi(x) > e$" for any x. Thereby $\mathfrak{c}_T = e$ works. ■

Chaitin has given many other variants of this proof. Another proof goes by the observation that if no such $\mathfrak{c}_T$ existed, then there would exist an unbounded lower bound function of the Kolmogorov complexity function.

2.2.3. *Other proofs.* Many other *model-theoretical* proofs of the incompleteness theorems have appeared.

In [Vop66], Vopěnka proved theorem 2.3 for Bernays-Gödel axiomatic set theory using Richard's paradox: "the least number not definable in 1000 words". More recently, in [Jec94], Jech gave a short proof of theorem 2.3 for set theory.

In [Kre68], Kreisel gave the first proof of theorem 2.3 using Henkin's arithmetized completeness theorem.

In [Boo89b, Boo89a], Boolos proved both theorems 2.2 and 2.3 using both model-theoretical techniques and Berry's paradox.

3. Incompleteness revisited

From Henkin's proof of the completeness theorem, one can derive the *arithmetized completeness theorem.* It is an important result that is essential for constructing arithmetical models and thus for proving Gödel's second incompleteness theorem.

The arithmetized completeness theorem asserts that any recursively axiomatizable consistent theory has an arithmetically definable model. We say that a formula ϕ in $\mathcal{L}_{\mathrm{PA}}$ *defines a model of* T *in a theory* S *in* $\mathcal{L}_{\mathrm{PA}}$ if we can prove within S that the set $\{\sigma : \sigma \text{ is a sentence in } \mathcal{L}_T \cup C \text{ that satisfies } \phi(\ulcorner\sigma\urcorner)\}$, where C is a set of new constants, forms an elementary diagram of a model of T with a universe from C.

Theorem 3.1 (Hilbert-Bernays arithmetized completeness theorem)**.** *There exists a* Δ_2 *formula* Tr_T *in* $\mathcal{L}_{\mathrm{PA}}$ *that defines a model of* T *in* $\mathrm{PA}+\mathrm{Cons}_T$*.*

The following is a corollary of this theorem: if $\mathcal{M}_0$ is a model of $\mathrm{PA}+\mathrm{Cons}_T$, then there exists a model $\mathcal{M}_1$ of T such that

(1) for any sentence ϕ in $\mathcal{L}_{\mathrm{PA}}$, $\mathcal{M}_1 \models \phi$ if and only if $\mathcal{M}_0 \models \mathrm{Tr}_T(\ulcorner\phi\urcorner)$,
(2) for any Σ_1 sentence ϕ in $\mathcal{L}_{\mathrm{PA}}$, if $\mathcal{M}_0 \models \phi$, then $\mathcal{M}_1 \models \phi$.

We then say that $\mathcal{M}_1$ is a model of T *definable* in a model of $\mathcal{M}_0$ of $\mathrm{PA}+\mathrm{Cons}_T$ and write $\mathcal{M}_1 \prec_d \mathcal{M}_0$.

3.1. Incompleteness in computability

3.1.1. *From* $\mathcal{K} = \{x : \varphi_x(x)\downarrow\}$ *or similar.* Using the same basic arguments we have used in the introduction, if we consider any non-recursive recursively enumerable set L, then given a Π_1-sound ($\equiv$ consistent) theory T, there is an $\mathfrak{n}_T^L \notin L$ such that T does not prove it. This is a form of the first incompleteness theorem.

Using the arithmetized completeness theorem 3.1, the second incompleteness theorem ($T \vdash 1\text{-}\mathrm{Cons}_T \rhd\!\neg\, \mathrm{Prov}_T(\mathrm{Cons}_T)$) can be proved as follows:

$\mathcal{K}$-related proof of a variant of theorem 2.3. We assume that Cons_T is derivable from T. Then by the completeness theorem, there exists a model $\mathcal{M}_0$ of T.

If $\mathcal{M}_0 \models \mathrm{Prov}_T(\ulcorner 0 \in L \urcorner)$, then let $\mathcal{M}_1 = \mathcal{M}_0$. Otherwise $\mathcal{M}_0 \models \neg\,\mathrm{Prov}_T(\ulcorner 0 \in L \urcorner)$ and thus $\mathcal{M}_0 \models \mathrm{Cons}_{T+0\notin L}$. Hence, by the Hilbert-Bernays arithmetized completeness theorem, there exists $\mathcal{M}_1 \prec_d \mathcal{M}_0$ such that

$$\text{either } \mathcal{M}_1 \models \mathrm{Prov}_T(\ulcorner 0 \in L \urcorner) \text{ (in case } \mathcal{M}_1 = \mathcal{M}_0) \text{ or } \mathcal{M}_1 \models \mathrm{Prov}_T(\ulcorner 0 \notin L \urcorner).$$

We iterate this construction.

Consider the final model $\mathcal{M}_{\mathfrak{n}_T^L}$, it satisfies $\neg\,\mathrm{Cons}_T$ by the previous form of the first incompleteness theorem: If we have $\mathcal{M}_{\mathfrak{n}_T^L} \models \mathrm{Prov}_T(\ulcorner \mathfrak{n}_T^L \in L \urcorner)$, then the 1-consistency of T is contradicted by the fact that $\mathfrak{n}_T^L \notin L$, since it is a Σ_1 statement. Hence, $\mathcal{M}_{\mathfrak{n}_T^L} \models \mathrm{Prov}_T(\ulcorner \mathfrak{n}_T^L \notin L \urcorner)$ which implies the non-consistency of T, since $\mathfrak{n}_T^L$ is such that $\mathfrak{n}_T^L \notin L$ is not provable in T. ■

3.1.2. *From computability functions.* In the sixties, Tibor Radó, a professor at the Ohio State University, thought of a simple non-computable function besides the standard halting problem for Turing machines. Given a fixed finite number of symbols and states, select those Turing machines which eventually halt when run with a blank tape. Among these programs, find the maximum number of non-blank symbols left on the tape when they halt. Alternatively, find the maximum number of time steps before halting. These functions are well-defined but uncomputable. Tibor Radó called them the Busy Beaver functions. For more on the Busy Beaver problem, read [Rad62, Lin63, LR65, Bra66, Bra83, Dew84, Dew85, Her88, MS90, MB90, LP07].

Alternative functions can be defined that are close in nature to these Busy Beaver functions. Let σ^{steps} be the function which to i gives the maximum number of steps for which a Turing machine with code $\leqslant i$ will keep running before halting starting with a blank tape. For a Turing machine M, t_M denotes the time complexity function of M: $t_M(x) = s$ if $M(x)$ halts after s steps. Following the Busy Beaver functions' definitions, we define σ^{value} to be the function which to i gives the maximum number which a Turing machine with code $\leqslant i$ will output, following a fixed convention, after halting starting with an input $\leqslant i$. These functions are in a sense inverses of the K_φ function.

Other functions can be defined following classical Kolmogorov complexity, *e.g.*, the function which to n gives the biggest number with Kolmogorov complexity lower than n.

We call these functions the σ functions. For each variant, we can define a function focusing on maximizing the *number of steps*, *e.g.*, σ^{steps}, or the *outputed values*, *e.g.*, σ^{value}. The value of one of the functions on a certain x is computable from x and the value of the other function on input $x + c$ for a certain constant c.

A result similar to Chaitin's result (see section 2.2.2) can be obtained concerning the σ functions:

Theorem 3.2 (Chaitin-like incompleteness theorem for σ functions)**.** *Let σ be one of the σ functions. Let T be an arithmetical-able consistent theory. There is a constant $\mathfrak{n}_T^\sigma$ such that*

$$T \vdash \mathrm{Cons}_T \rhd \forall s \neg\,\mathrm{Prov}_T(\ulcorner \sigma(\mathfrak{n}_T^\sigma) < s \urcorner). \tag{3.1}$$

Proof. Consider a Π_1 formula ϕ_σ in the language of T such that $\phi_\sigma(x, s)$ expresses that $\sigma(x) < s$.

Working in T, for a given x, take the smallest s such that $\mathrm{Prov}_T(\ulcorner \phi_\sigma(x,s)\urcorner)$ holds. T being consistent and ϕ_σ Π_1, $\phi_\sigma(x,s)$ also holds.

$\mathrm{Prov}_T(\ulcorner \phi_\sigma(x,s)\urcorner)$ is a Σ_1 formula and thus can be seen as $\exists y \psi(x,s,y)$ or equivalently $\exists\langle s,y\rangle \psi(x,s,y)$ where ψ is Δ_0.

Thus there is a Turing machine computing ψ. Consider its code i_ψ (or its number of states or transitions, depending on the choice of σ). For large enough x, *i.e.*, $x > i_\psi + c$, knowing that $\phi_\sigma(x,s)$ holds (using the computation through shifting, *i.e.*, the constant c, between both types of σ functions), we know that $\sigma(x) < s$ and thus there is an $s' = \langle s'_1, s'_2\rangle < s$ such that $\psi(x,s'_1,s'_2)$ holds. But for each $s' = \langle s'_1, s'_2\rangle$ smaller than s, the statement $\neg\psi(x,s'_1,s'_2)$ is true by the minimality of s, and provable (being Δ_0). Thus we have $\neg\,\mathrm{Cons}_T$. ∎

It is also possible to go through the same proof but using Kolmogorov functions (classical Kolmogorov complexity function C, K_φ, $\ldots$) and a variant of Chaitin's theorem 2.6, following closely the proof of theorem 3.2: Let K be a Kolmogorov function. If T is consistent, then there exists $\mathfrak{n}_T^K$ such that for all x, $T \nvdash K(x) > \mathfrak{n}_T^K$. Moreover, if T is ω-consistent, then for all x, if $K(x) > \mathfrak{n}_T^K$, then $T \nvdash K(x) \leqslant \mathfrak{n}_T^K$.

From there, we can show Gödel's second incompleteness theorem in a model-theoretical way.

Model-theoretical proof of theorem 2.3 using σ functions. We have supposed that T is consistent. So, let $\mathcal{M}_0$ be a model of T.

If $\mathcal{M}_0 \models \mathrm{Prov}_T(\ulcorner \forall x\ \neg t_{T_0}(0) = x\urcorner)$, then let $\mathcal{M}_1 = \mathcal{M}_0$. Otherwise,

$$\mathcal{M}_0 \models \neg\,\mathrm{Prov}_T(\ulcorner \neg\exists x\ t_{T_0}(0) = x\urcorner)$$

Thereby, $\mathcal{M}_0 \models \mathrm{Cons}_{T+\exists x\ t_{T_0}(0)=x}$. Hence there exists, by theorem 3.1, $\mathcal{M}_1 \prec_d \mathcal{M}_0$ such that either $\mathcal{M}_1 \models \mathrm{Prov}_T(\ulcorner \neg\exists x\ t_{T_0}(0) = x\urcorner)$ (when $\mathcal{M}_1 = \mathcal{M}_0$), or $\mathcal{M}_1 \models \exists x\,\mathrm{Prov}_T(\ulcorner t_{T_0}(0) = x\urcorner)$ (because $\{(i,x) : t_{T_i}(0) = x\}$ is Δ_0, in other words primitive recursive, and thus its truth in $\mathcal{M}_1$ implies its provability).

We iterate this construction (consider now $\mathcal{M}_1$ and "$t_{T_1}(0) = x$", instead of $\mathcal{M}_0$ and "$t_{T_0}(0) = x$"; in i-th iteration, consider $\mathcal{M}_i$ and "$t_{T_i}(0) = x$").

Consider the last model $\mathcal{M}_{\mathfrak{n}_T^\sigma}$, the model constructed after the $\mathfrak{n}_T^\sigma$-th iteration. This model satisfies $\forall i \leqslant \mathfrak{n}_T^\sigma\ \mathrm{Prov}_T(\ulcorner \neg\exists x\ t_{T_i}(i) = x\urcorner) \vee \exists x\,\mathrm{Prov}_T(\ulcorner t_{T_i}(i) = x\urcorner)$ by theorem 3.1. In this model, for each $i \leqslant \mathfrak{n}_T^\sigma$ such that the second case holds ($\exists x\,\mathrm{Prov}_T(\ulcorner t_{T_i}(i) = x\urcorner)$), we take the smallest appropriate x and choose s to be greater than all these x's. Thus, this model satisfies the provability in T of $\sigma(\mathfrak{n}_T^\sigma) < s$ and thus satisfies $\neg\,\mathrm{Cons}_T$ by (3.1). ∎

This argument can be carried out for other functions than σ. In particular for the variants of the Busy Beaver functions.

The second incompleteness theorem 2.3 can also be proved in this manner from the above Kolmogorov function variant of theorem 3.2.

It is an open question to carry out this type of argument (for proving both incompleteness theorems) for bizarre functions derived from the Busy Beaver functions, defined in [LP07], *e.g.,* consider the function giving the parity of one of the Busy Beaver functions. One of the obvious missing properties of these functions is unboundedness.

3.1.3. *Giving its own relative consistency.* We say that a statement ϕ is a *revelation for* T if ϕ is unprovable in T and its consistency relative to T (if T is consistent, so is $T + \phi$) is provable from itself in T:

$$T \vdash \phi \rhd \mathrm{Cons}_T(\phi)$$

The links between incompleteness and computability functions described above in section 3.1.2 have yielded the following serendipitous result.

Theorem 3.3 (Serendipitous incompleteness theorem for σ functions)**.** *Let σ be one of the σ functions. If T is consistent, then there exists a natural number $\mathfrak{r}_T^\sigma$ such that for all x, $\sigma(\mathfrak{r}_T^\sigma) < x$ is a revelation for T.*

Proof. Consider the Π_1 statement $\forall x\ \psi(x)_x$ equivalent to $\mathrm{Cons}_{T+\phi}$.

$\psi \in \Delta_0$ and thus there is a machine M_ψ with code i_ψ such that M_ψ decides $\{x : \psi(x)_x\}$: M_ψ on input x eventually enters an acceptance state if $\psi(x)_x$, or a rejection state otherwise.

Consider another Turing machine M'_ψ which runs M_ψ successively on each natural number starting from 0 and stops and writes the counter example of ψ if the simulation of M_ψ enters a rejection state.

Let i'_ψ be the code of Turing machine M'_ψ. $\sigma(i'_\psi)$ makes the verification of $\forall x\ \psi(x)_x$ a Δ_0 property.

By using Kleene's recursion theorem on this previous construction, we find $\mathfrak{r}_T^\sigma$ such that knowing (or bounding) the value of $\sigma(\mathfrak{r}_T^\sigma)$ makes the verification of $\mathrm{Cons}_{T+\sigma(\mathfrak{r}_T^\sigma)\leqslant x}$ a Δ_0 property. Knowing that T is consistent and assuming $\sigma(\mathfrak{r}_T^\sigma) \leqslant x$, T thus proves $\mathrm{Cons}_T(\sigma(\mathfrak{r}_T^\sigma) \leqslant x)$.

By Gödel second incompleteness theorem, $\sigma(\mathfrak{r}_T^\sigma) \leqslant x$ is an unprovable statement in T. ■

3.2. Interpretations of Chaitin's theorem

Chaitin's famous version of Gödel's first incompleteness theorem (see section 2.2.2) is compelling for various obvious reasons. Firstly a statement of the type "the Kolmogorov complexity of this integer is greater than that integer" looks more mathematically natural than a consistency statement and secondly it gives a bound on the provable complexity of objects in a given theory. The question that arises forthrightly is the relevance of this bound to measure the *complexity*, the *power*, or *information content* of a theory.

We will now discuss the validity of the common way of interpreting Chaitin's theorem. Many people have addressed criticisms towards this interpretation. In particular see [Fal96, Raa98]. We try to sum up these criticisms here.

Chaitin's result, theorem 2.6, has been interpreted to show that in a formalized theory one cannot prove an object to be more complex than the complexity of the theory itself. This received interpretation claims that the limiting constant $\mathfrak{c}_T$ is determined by the complexity of the theory T itself and is a good measure of the strength of the theory.

As Chaitin puts it in [Cha82]: "I would like to measure the power of a set of axioms and rules of inference. I would like to be able to say that if one has ten pounds of axioms and a twenty-pound theorem, then that theorem cannot be derived from those axioms."

It is assumed here that the algorithmic complexity of the axioms gives a good measure of the *power*, or *information content*, of the theory. The constant $\mathfrak{c}_T$ is assumed to depend on the complexity of the axioms of T. The finite bound given by the constant $\mathfrak{c}_T$ is hence thought to reflect the *power*, or *information content*, of the theory.

By playing with Kleene's fixed point theorem, for any suitable theory T, one can construct acceptable enumerations of partial recursive functions yielding constants $\mathfrak{c}_T$ equal to 0 or arbitrarily large, whatever the theory T.

A closer inspection shows that the value of $\mathfrak{c}_T$ is actually determined simply by the smallest (by its code) Turing machine which does not halt, but for which this cannot be proved in T. It is really hard to see why the code of such a Turing machine would reveal anything interesting about the *power* or *information content* of T.

Considering a strong theory like ZFC, *Zermelo-Fraenkel set theory with the axiom of choice*, we could compare its constant $\mathfrak{c}_{\mathrm{ZFC}}$ to the constant of a weak theory, say PA, *Peano Arithmetic.* The constants depend on our acceptable enumeration of partial recursive functions. Thus, suppose we have $\mathfrak{c}_{\mathrm{ZFC}} > \mathfrak{c}_{\mathrm{PA}}$. We can then add to *PA* all true sentences of the form $\neg\exists x\ \varphi_e(0) = x$ which are provable in ZFC, for all $e < \mathfrak{c}_{\mathrm{ZFC}}$. It follows from a result of Kreisel and Levy [KL68] that this new theory cannot possibly come even close to the power of ZFC. But the constants of this new theory and ZFC are now equal, and hence, they should, according to the received interpretation, have the same *power.* Furthermore, we may still add to our new theory one more true sentence $\neg\exists x\ \varphi_{\mathfrak{c}_{\mathrm{ZFC}}}(0) = x$. Now the constant of this theory is bigger than the one of ZFC. This whole argumentation shows that one has to be careful on the interpretation given to the constants $\mathfrak{c}_T$'s.

As mentioned in [Fal96], the only thing that these constants could at most tell of a theory is what propositions of the form "$K(\cdot) > \cdot$" it can prove. Therefore, withstanding all the above arguments, one could wonder whether adding as an axiom a sentence of the form "$K(x) > c$" could not be equivalent to the relative 1-consistency of a strong (consistency-wise) unprovable statement or even to the 1-consistency of a theory. For example, the 1-consistency of a large cardinal axiom[5] would also only add "information" about some of the propositions and be incredibly weaker than the large cardinal axiom itself. This would give credit to Chaitin's interpretation of his theorem and present the constant $\mathfrak{c}_T$ as a *partial* measure of the *power* of a theory T.

Having a link between consistency (or soundness) and computability ($\mathcal{K}$, Busy beaver functions or Kolmogorov complexity) would make possible an understanding of what properties consistency adds to a theory. Adding consistency as an axiom would then yield new combinatorial properties. Until now, consistency has been seen as a strange statement, only considered because of Gödel's second incompleteness theorem. It is true that even if one can construct stronger theories by adding as a new axiom its consistency, it is not clear in what way the obtained theory is stronger. It could well be that the only additional *information* this new theory has is this consistency statement and that nothing else is added because of this additional axiom. In fact, as mentioned in the introduction of this paper, one would need to *add*[6] $\omega^{\omega^{\omega+1}}$ times a reflection principle to our theory to cover all true arithmetic statements.

Taking into account the above arguments, we see that the only *total* measure one could get of a theory through Chaitin's theorem 2.6 would not be a constant $\mathfrak{c}_T$ but a set $\mathcal{C}_T$ of constants for which Chaitin's proposition is unprovable in T. If we want the above objections not to apply (in particular modifying by Kleene's fixed point theorem the

[5]See [KM78, Kan94].

[6]For any uniform reflection progression $\{T_a\}_{a\in\mathbf{O}}$, there is a branch B in an ordinal notation system $\mathbf{O}$, such that there is, for any true arithmetical sentence ϕ, an a in B with $|a| < \omega^{\omega^{\omega+1}}$ for which ϕ is provable in T_a. For details, see [Fef62, FS62].

acceptable enumeration with which we work), $\mathcal{C}_T$ should necessarily be infinite and non-recursive. This set could not then be used to form an additional axiom if we want our theory to stay recursively axiomatizable.

References

[Boo89a] Boolos (G.), « A letter from George Boolos », *Notices of the American Mathematical Society*, vol. 36, 1989, p. 676.

[Boo89b] Boolos (G.), « A new proof of the Gödel incompleteness theorem », *Notices of the American Mathematical Society*, vol. 36, 1989, p. 383–391.

[Boo95] Boolos (G.), *The Logic of Provability.* Cambridge University Press, Cambridge, 1995.

[Bra66] Brady (A. H.), « The conjectured highest scoring machines for Rado's $\sigma(k)$ for the value $k = 4$ », *IEEE Transactions on Elec. Comput.*, vol. EC-15, 1966, p. 802–803.

[Bra83] Brady (A. H.), « The determination of the value of Rado's noncomputable function $\sigma(k)$ for four-state Turing machines », *Mathematics of Computation*, vol. 40, 1983, p. 647–665.

[Cha82] Chaitin (G.), « Gödel's theorem and information », *International Journal of Theoretical Physics*, vol. 22, 1982, p. 941–954.

[Dev84] Devlin (K. J.), *Constructibility.* Springer, 1984.

[Dew84] Dewdney (A. K.), « Computer recreations: A computer trap for the busy beaver, the hardest-working Turing machine », *Scientific American*, vol. 251, n° 2, August 1984, p. 19–23.

[Dew85] Dewdney (A. K.), « Computer recreations », *Scientific American*, vol. 252, n° 4, April 1985, p. 12–16.

[End72] Enderton (H. B.), *A Mathematical Introduction to Logic.* Academic Press, New York, 1972.

[Fal96] Fallis (D.), « The source of Chaitin's incorrectness », *Philosophia Mathematica*, vol. 3, n° 4, 1996, p. 261–269.

[Fef60] Feferman (S.), « Arithmetization of metamathematics in a general setting », *Fundamenta Mathematicae*, vol. 49, 1960, p. 35–92.

[Fef62] Feferman (S.), « Transfinite recursive progressions of axiomatic theories », *Journal of Symbolic Logic*, vol. 27, n° 3, 1962, p. 259–316.

[FS62] Feferman (S.) et Spector (C.), « Incompleteness along paths in progressions of theories », *Journal of Symbolic Logic*, vol. 27, n° 4, December 1962, p. 383–390.

[Göd31] Gödel (K.), « Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I », dans *Monatshefte für Mathematik und Physik* [SFKM$^+$86], p. 173–198.

[HB39] Hilbert (D.) et Bernays (P.), *Grundlagen der Mathematik*, vol. 2. Springer, 1939.

[Hen57] Henkin (L.), « A generalization of the concept of ω-completeness », *Journal of Symbolic Logic*, vol. 22, n° 1, March 1957, p. 1–14.

[Her88] Herken (R.), *The Universal Turing Machine: A Half-Century Survey.* Oxford University Press, Oxford, England, 1988.

[Hod93] Hodges (W.), *Model theory*, vol. 42 (coll. *Encyclopedia of Mathematics and its Applications*). Cambridge University Press, Cambridge, 1993.

[Jec78] Jech (T.), *Set Theory.* Academic Press, New York, 1978.

[Jec94] Jech (T.), « On Gödel's second incompleteness theorem », *Proceedings of the American Mathematical Society*, vol. 121, 1994, p. 311–313.

[Kan94] Kanamori (A.), *The Higher Infinite.* Springer Verlag, 1994.

[Kay91] Kaye (R.), *Models of Peano arithmetic*, vol. 15 (coll. *Oxford Logic Guides*). Oxford University Press, Oxford, 1991.

[KL68] Kreisel (G.) et Lévy (A.), « Reflection principles and their use for establishing the complexity of axiomatic systems », *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, vol. 14, 1968, p. 97–142.

[Kle52] Kleene (S.), *Introduction to Metamathematics.* North-Holland Publishing, New York, 11th edition édition, 1952.

[KM78] Kanamori (A.) et Magidor (M.), « The evolution of large cardinal axioms in set theory », dans Muller (G. H.) et Scott (D. S.), éditeurs, *Higher Set Theory*, vol. 669 (coll. *Lecture Notes in Mathematics*), p. 99–275. Springer Verlag, Berlin, 1978.

[Kot94] Kotlarski (H.), « On the incompleteness theorems », *Journal of Symbolic Logic*, vol. 59, n° 4, 1994, p. 1414–1419.

[Kot96] Kotlarski (H.), « An addition to Rosser's theorem », *Journal of Symbolic Logic*, vol. 61, n° 1, 1996, p. 285–292.

[Kot98] Kotlarski (H.), « Other proofs of old results », *Mathematical Logic Quarterly*, vol. 44, 1998, p. 474–480.

[Kot04] Kotlarski (H.), « The incompleteness theorems after 70 years », *Annals of Pure and Applied Logic*, vol. 126, 2004, p. 125–138.

[Kre50] Kreisel (G.), « Notes on arithmetical models for consistent formulae of the predicate calculus », *Fundamenta Mathematicae*, vol. 37, 1950, p. 265–285.

[Kre68] Kreisel (G.), « A survey of proof theory », *Journal of Symbolic Logic*, vol. 33, 1968, p. 321–288.

[Laf02] Lafitte (G.), *Calculs et Infinis*. PhD thesis, École Normale Supérieure de Lyon, decembre 2002.

[Lin63] Lin (S.), *Computer Studies of Turing Machine Problems*. PhD thesis, The Ohio State University, Colombus (Ohio), 1963.

[Löb55] Löb (M. H.), « Solution of a problem of leon henkin », *Journal of Symbolic Logic*, vol. 20, 1955, p. 115–118.

[LP07] Lafitte (G.) et Papazian (C.), « The fabric of small Turing machines », dans Cooper (S. B.), Kent (T. F.) et Benedikt Löwe (A. S.), éditeurs, *Computation and Logic in the Real World, Third Conference of Computability in Europe, CiE 2007*. 2007.

[LR65] Lin (S.) et Radó (T.), « Computer studies of Turing machine problems », *Journal of the Association for Computing Machinery*, vol. 12, n° 2, April 1965, p. 196–212.

[LV90] Li (M.) et Vitányi (P.), *Handbook of Theoretical Computer Science*, chap. Kolmogorov complexity and its applications, p. 187–254. Amsterdam, Elsevier, 1990.

[MB90] Marxen (H.) et Buntrock (J.), « Attacking the busy beaver 5 », *Bulletin of the EATCS*, vol. 40, 1990, p. 247–251.

[MS90] Machlin (R.) et Stout (Q. F.), « The complex behavior of simple machines », *Physica*, vol. 42D, 1990, p. 85–98.

[Odi89] Odifreddi (P.), *Classical Recursion Theory*. North Holland Publishing, 1989.

[Odi99] Odifreddi (P.), *Classical Recursion Theory*, vol. Vol. II. North Holland Publishing, 1999.

[Oll08] Ollinger (N.). « Universalities in cellular automata ». personal communication (submitted to JAC 2008), 2008.

[Raa98] Raatikainen (P.), « On interpreting Chaitin's incompleteness theorem », *Journal of Philosophical Logic*, vol. 27, 1998, p. 569–586.

[Rad62] Radó (T.), « On non-computable functions », *Bell System Technical Journal*, vol. 41, May 1962, p. 877–884.

[Rog58] Rogers (H.), « Gödel numberings of partial recursive functions », *Journal of Symbolic Logic*, vol. 23, 1958, p. 331–341.

[Rog67] Rogers (H.), *The Theory of Recursive Functions and Effective Computability*. MIT Press, 1967.

[Ros36] Rosser (J. B.), « Extensions of some theorems of Gödel and Church », *Journal of Symbolic Logic*, vol. 1, 1936, p. 87–91.

[SFKM$^+$86] S. Feferman (W. D.), Kleene (S.), Moore (G.), Solovay (R.) et van Heijendort (J.), éditeurs, *Kurt Gödel: Collected Works*, vol. 1. Oxford University Press, Oxford, 1986.

[Sho67] Shoenfield (J. R.), *Mathematical logic*. Addison-Wesley, Reading, Massachusetts, 1967.

[Smo77] Smorynski (C.), *Handbook of mathematical logic*, chap. The incompleteness theorems, p. 821–865. Amsterdam, North-Holland, 1977.

[Smu93] Smullyan (R. M.), *Recursion Theory for Metamathematics*. Oxford University Press, New York, 1993.

[Vop66] Vopěnka (A.), « A new proof of the Gödel's result of non-provability of consistency », *Bulletin de l'Académie Polonaise des Sciences*, vol. 14, n° 3, 1966, p. 111–116.

[VS03] Vereshchagin (N.) et Shen (A.), *Computable Functions*. American Mathematical Society, 2003.

Journées Automates Cellulaires 2008 (Uzès), pp. 90-101

NEIGHBORHOOD TRANSFORMATIONS ON GRAPH AUTOMATA

BRUNO MARTIN AND CHRISTOPHE PAPAZIAN

Université de Nice–Sophia Antipolis, I3S, UMR 6070 CNRS, 2000 route des Lucioles, BP 121, F-06903 Sophia Antipolis Cedex.
E-mail address: {Bruno.Martin|Christophe.Papazian}@unice.fr

Abstract. We consider simulations of graph automata. We introduce two local transformations on the neighborhood: splitting and merging. We explain how to use such transformations, and their consequences on the topology of the simulated graph, the speed of the simulation and the memory size of simulating automata in some cases. As an example, we apply these transformations to graph automata embedded on surfaces and we link our results with some simulation results between cellular automata on Cayley graphs.

1. Introduction

In this paper, we consider simulations between networks of automata arranged on graphs which are embedded on surfaces. The way to draw graphs on surfaces comes from combinatorial topology, an older name for algebraic topology which was addressed by Kuratowski [4].

Combinatorial topology (see [3]) was developed at first as a branch of geometry. The work of Euler and a number of nineteenth-century geometers on polyhedra is part of the development. Under the scope of this theory is also the study of surfaces. Surfaces are topological spaces in which every point has a neighborhood that is topologically equivalent to an open disk. The simplest example of a surface is the plane. Other objects can be constructed in a combinatorial way by gluing disks together. With this kind of operation one gets a cylinder, a surface with boundary, or the torus which is the surface that results when both pairs of opposite sides of a rectangle are identified.

This kind of networks of automata has to be compared with cellular automata on Cayley graphs. Both models share the same underlying networks but are described in a completely different fashion. Instead of drawing the graph of the network on a surface, it is defined by the Cayley graph of a finitely presented group. The approach, more algebraic, brings more constraints. Some authors already considered simulations between cellular automata on Cayley graphs: Róka [10, 11, 12, 13] proposed different simulations extended by Martin [5, 6, 7].

Key words and phrases: Graph automata, Cellular automata on Cayley graphs, algorithmics, topology.
(This work has been supported by the Interlink/MIUR project "Cellular Automata: Topological Properties, Chaos and Associated Formal Languages" and by the french ANR programme Sycomore.)

Some results can be imported from the Cayley graphs approach to the combinatorial topology approach and we will discuss their similarities and differences.

The paper is organized as follows. We introduce our model of computation in Section 2. Section 3 presents our local transformations on the neighborhood and gives some examples. Section 4 uses the local transformations to simulate finite graph automata embedded on surfaces.

2. Notation and definitions

We start with two surfaces with boundary: the *cylinder* and the *Moebius strip* (see Fig. 1). The *cylinder* which can be described as a square in which top and bottom edges

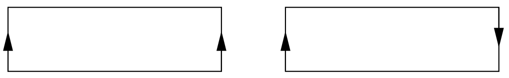


Figure 1: Two surfaces with boundary; a cylinder (left) and a Mœbius strip (right).

are given parallel orientations and the left and right edges are joined to place the arrow heads and tails into coincidence. The Möbius strip is a one-sided non-orientable surface obtained by cutting a closed band into a single strip, giving one of the two ends thus produced a half twist, and then reattaching the two ends.

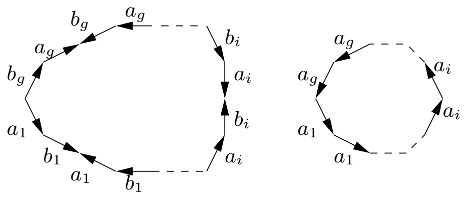


Figure 2: Drawing of orientable surface (left) and non-orientable surface (right).

We then consider "classical" surfaces for drawing a picture of a graph (cf. section 2.1). There are two types of surfaces: *orientable* and *non-orientable.* The orientable surface of genus 0 and 1 are respectively called a *plane* and a *torus.* When we increase the genus g of the orientable surface for $g \geq 2$, we obtain $\overrightarrow{S_g}$ a g-handled torus. The non-orientable surfaces of genus 1 and 2 are respectively a *projective plane* and a *Klein bottle.* Fig. 2 left describes an orientable surface of genus g in which each pair of edges sharing the same label are joined together. Fig. 2 right describes S_g, an non-orientable surface of genus g.

In the rest of the paper we will keep the notations of combinatorial topology for the surfaces we consider. Thus, the usual ring becomes a cylinder (abbreviated by *Cyl*), a "usual" torus remains a torus (abbreviated by $\overrightarrow{S}_1$).

2.1. Embedding graphs on surfaces

A *graph* G is an ordered pair $G = (V, E)$ where V is a set of *vertices* (or *nodes*) and E is a set of *edges* which are pairs of distinct vertices. A *path* is a sequence of vertices, each adjacent to the next. A *cycle* is a path with at least 3 vertices such that the last vertex is adjacent to the first. Given x and y two vertices of G, the *distance* between them is the length of a minimal path from x to y.

Since our goal is to embed regular graphs (i.e. isomorphic to Cayley Graphs) on surfaces and to associate a finite state machine to each vertex of the graph, we need some further definitions on graphs.

A graph is *planar* if it can be drawn in the plane so that no edges intersect or, equivalently, if it can be *embedded* in the plane. A nonplanar graph cannot be drawn without edge intersections. More generally, we consider in this paper graphs which are *embeddable* on an orientable surface $\overrightarrow{S_g}$, that is which can be drawn on $\overrightarrow{S_g}$ without crossing edges.

When a graph is drawn without any crossing, any cycle that surrounds a region without any edge reaching from the cycle inside to such region forms a *face*, including the outer, infinitely large regions (when existing). Observe that the notion of face is independent from the embedding [2].

The *dual* of a given planar graph G has a vertex for each face of the graph and an edge for each edge joining two neighboring regions. Fig. 3 illustrates the embedding of an hexagonal grid on a torus and the embedding of its dual on a torus. Vertices with the same number have to be identified as well as edges joining identical vertices.

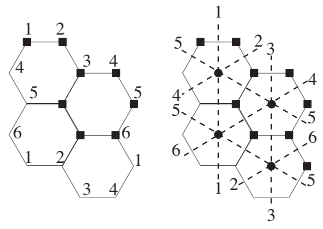


Figure 3: Dual embeddings of an hexagonal graph on a torus. On the left, vertices with the same number are identified like on the right, the dotted edges with the same numbers.

2.2. Graph automata

A *partitioned graph automaton* (PGA for short) over a graph G is a 4-tuple $\mathcal{A} = (Q, G, N, \delta)$ for which we associate a finite state machine called a *cell* to each vertex of the graph G. The set Q denotes the finite set of the states, $G = (V, E)$ is a graph, N the neighborhood (including the cell itself and nodes at distance 1 together with a local numbering as described by Fig. 4; the numbering gives an ordering of the neighbors that will be used by the local transition function) and $\delta : Q^{\sharp N} \rightarrow Q^{\sharp N}$ is the local transition function which updates the state of cell i at time t according to the states of (copies of) its neighbors at time $t - 1$, analogously with the *partitioned CA* (PCA for short) introduced in [8]. In a PGA (as well as in a PCA), the states are partitioned according to the neighborhood and only the relevant pieces of states are available to any state. Sub-states are gathered to form only one state to be updated. That is, each state of a PGA is a $\sharp N$-tuple, each tuple contains information for a specific neighbor. This model simplifies the simulations we are considering in section 2.3. We define a distinguished state q, the *quiescent state* that verifies $\delta(q,\dots,q) = q^{\sharp N}$. Note that we only consider the radius 1 neighborhood (of one cell) defined as the set of vertices at distance at most one from the cell (thus including the cell itself) that we depict on Fig 4. The k-neighborhood (of a cell) is the set of vertices at distance at most k from the cell. Hence, as we need to know the neighborhood of a cell to compute one transition step, we need to know the k-neighborhood to compute k transition steps. We define a *configuration* of the PGA as an application c which attributes a state to each cell. The set of all the configurations of a PGA is denoted by $\mathbb{C} = Q^{\sharp N \sharp V}$ on which the *global function* Δ of the PGA is defined by applying globally the local transition function.

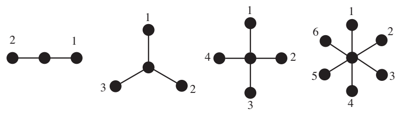


Figure 4: Different kinds of neighborhoods; from left to right: N_2, N_3, N_4 and N_6.

Definition 2.1 gives the formal statement of a simple PGA embedded on a cylinder and the behavior of the transition function is depicted on Fig. 5.

Definition 2.1. A 2-neighbor PGA on a cylinder is $\mathcal{A} = (Q, G, N_2, \delta)$ with set of states $Q = L \times C \times R$. L, C and R are all non-empty subsets of the same set of states Q' with L the set of left internal states, C the set of center internal states and R the set of right internal states, $G = C_n$ (the cycle graph with n vertices), N_2 the von Neumann neighborhood, and δ the local transition function:

$$\delta : R \times C \times L \to L \times C \times R$$

A configuration of $\mathcal{A}$ is a mapping $\mathbb{Z}_n \to L \times C \times R$. The set of all configurations is denoted by $\mathbb{C}$. We denote by LEFT (CENTER, RIGHT resp.) the projection function which picks out the left (resp. center, right) element of a triple in $L \times C \times R$. The global function is $\Delta(c)(i) = \delta\left(\textsc{right}(c(i-1)), \textsc{center}(c(i)), \textsc{left}(c(i+1))\right)$ where $i, i-1$ and $i+1$ are integers modulo n.

Definition 2.1 can be easily adapted to the neighborhoods depicted on Fig. 4.

In the sequel, we will consider some particular drawings of graphs on surfaces for which we introduce some notation. The first letter(s) denotes the surface on which the graph will be embedded with the subscript denoting its genus (if relevant). The second letter gives the neighborhood of the graph according to Fig. 4. Last parameter gives the number of vertices of the graph. The simplest one is the embedding of a cycle graph with n vertices on a cylinder (Fig. 5). It will be denoted by Cyl$N_2(n)$. Next is the toroidal mesh which is the embedding of the cartesian sum[1] of two cycle graphs with respectively m and n vertices on a torus. It will be denoted by $\overrightarrow{S_1}N_4(m,n)$. We also consider the embedding of an hexagonal graph (resp. triangle, its dual graph) on a torus denoted by $\overrightarrow{S_1}N_6(m,n)$ (resp. $\overrightarrow{S_1}N_3(m,n)$). Observe that, for simplicity reason, the parameters of $\overrightarrow{S_1}N_3(m,n)$ are the same than $\overrightarrow{S_1}N_6(m,n)$. Indeed, we count the number of hexagons in each principal

[1]The cartesian sum is often called cartesian product but the definitions differ [2].

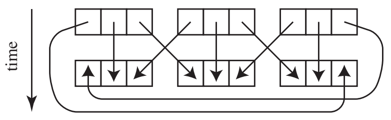


Figure 5: Two configurations of a 2-neighbor 3 cells PGA embedded on a cylinder Cyl$N_2(3)$.

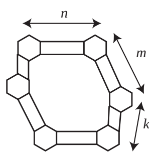


Figure 6: S_3N_6: embedding of an hexagonal graph in a 3-handled torus.

direction. We will also consider a generalization as represented in Fig. 6: the embedding of an hexagonal graph (resp. triangle, its dual graph) on S_3 (the non-oriented surface of genus 3) denoted by $S_3N_6(m,n,k)$ (resp. $S_3N_3(m,n,k)$). The regularity and extendability of $S_3N_6(m,n,k)$ comes from its group representation and was considered by Róka in her work on cellular automata on Cayley graphs.

Since we are dealing with graphs which are (cellularly) embedded into surfaces, there is no need here to fully define the interconnection pattern. All the graphs we consider being regular, the way they are connected is defined by the neighborhood (cf. Fig. 4).

2.3. Simulation

Below, we propose the definition of a step by step simulation between two PGAs. It expresses that if a PGA A simulates each step of PGA B in τ units of time, there must exist effective applications between the corresponding configurations:

Definition 2.2. Let $\mathbb{C}_A$ and $\mathbb{C}_B$ be the two sets of PGA configurations A and B. We say that A simulates each step of B in time τ (and we note $B \overset{\tau}{\prec} A$) if there exists a constant $\tau \in \mathbb{N}$ and two recursive functions $\kappa : \mathbb{C}_B \to \mathbb{C}_A$ and $\rho : \mathbb{C}_A \to \mathbb{C}_B$ such that $\kappa \circ \rho = \mathrm{Id}$ and for all $c, c' \in \mathbb{C}_B$, there exists $c'' \in \mathbb{C}_A$ such that if $c' = \Delta_B(c)$, $c'' = \Delta_A^{\tau}(\kappa(c))$ with $\rho(c'') = c'$, where Δ_M denotes a global transition of PGA M and Δ_M^t the t-th iterate of a global transition of PGA M.

Depending upon the value of τ, we say that the simulation is *elementary* if $\tau = 1$, *simple* if $\tau = O(1)$ and *general* for $\tau = O(f(c))$ with f denoting any given time-complexity function on c, the size of the input.

3. Neighborhood transformations

Neighborhood transformations can be seen as local transformations. Such transformations allow to handle the same computation with slightly different automata, without changing the major topological properties of the GA. We present two local transformations: *splitting* and *merging* and we give some consequences on the computations.

3.1. Homogeneous GAs and splittings

Homogeneous GAs are networks where all the vertices have the same number of neighbors. No other assumption is made. We first introduce the *splitting.*

Definition 3.1. A splitting s_G is a local transformation that replaces simultaneously each single vertex by a subgraph G with the same number of outgoing edges (edges that do not belong to the subgraph but that link it to the network). The splitting is regular if the GA remains homogeneous (if G is homogeneous).

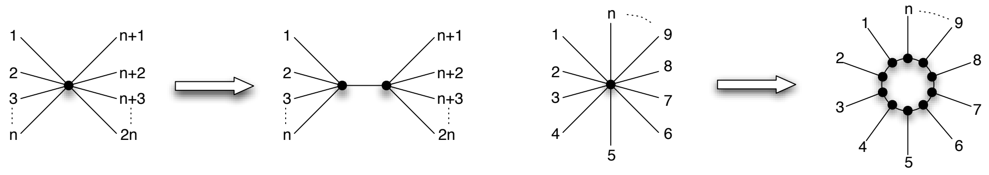


Figure 7: 2-split and multisplit.

Fig. 7 shows two simple regular splits. The 2-split transforming a $2n$-node into two $n+1$-nodes and the multisplit transforming a n-node into n 3-nodes.

If we consider simulations, we obtain Lemma 3.2.

Lemma 3.2. *Any GA $\mathcal{N}$ can be simulated by any other GA $s_G(\mathcal{N})$ obtained by application of a split. The bound on the factor of deceleration equals one plus the diameter of the subgraph G; with our notation, $\mathcal{N} \overset{d+1}{\prec} s_G(\mathcal{N})$.*

Proof. The diameter of a graph is the maximum length of shortest paths between any two vertices of the graph. Obviously, each subgraph G needs one step of computation to "read" the states of neighbor subgraphs. Then, d steps are required to obtain, in each nodes of the subgraphs the complete information on the neighborhood to compute the simulated transition. Hence one can compute a simple simulation with a slowdown factor of $d+1$. ■

Hence, we can simulate complex homogeneous GAs with a high degree of connectivity with bigger but less connected GAs.

3.2. Example of a 2-split

Lemma 3.3 explains the simulation of any 6-neighbor PGA by a 4-neighbor PGA.

Lemma 3.3. *N_6-PGA $\overset{2}{\prec}$ N_4-PGA.*

The idea is to cut the hexagon using a 2-split for transforming a 6 node into two 4 nodes and by adding a new part state called a *layer* (R for the left part and L for the right part). The simulation depicted on Fig. 9 is as follows:

(1) gather the missing neighbors information; pack it in the layer (Fig. 9 left);
(2) simulate one step of h according to the new neighborhood (Fig. 9 right).

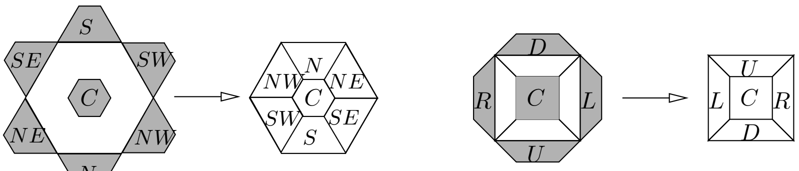


Figure 8: Transition of a N_6-PGA (left) and of a N_4-PGA (right).

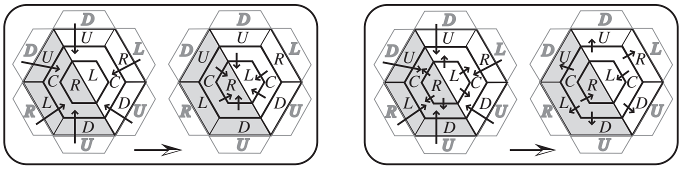


Figure 9: Simulating N_6-PGA by N_4-PGA: gather neighbor informations (left) and simulate one transition step (right).

Proof. A N_6-PGA [9] is (Q, G_6, N_6, h) with $Q = C \times N \times NE \times SE \times S \times SW \times NW$ sets of center, north, north-east, south-east, south, south-west and north-west part states. G_6 is the graph of hexagons which is embedded on a surface, N_6 is as depicted on Fig 4. The local function h is a mapping (see Fig. 8):

$$h : C \times S \times SW \times NW \times N \times NE \times SE \to C \times N \times NE \times SE \times S \times SW \times NW$$

A N_4-PGA is (Q, G_4, N_4, σ) with $Q = (C, U, R, L, D)$ sets of center, up, right, left and down part states. G_4 is a toroidal mesh, N_4 as on Fig 4. The local function σ is a mapping: $\sigma : C \times D \times L \times U \times R \to C \times U \times R \times D \times L$ (Fig. 8 right).

To simulate a N_6-PGA by a N_4-PGA, we distinguish periodically two cells: one which gathers the contents of the right part of the hexagon and, respectively, one which gathers the left part of the hexagon. The first rule of σ is:

$$\begin{array}{llll} (1) & (C, D, L, U, R) \mapsto & (C, U, R = (D, U, R), D, L) & \text{gathering left part} \\ (2) & (C, D, L, U, R) \mapsto & (C, U, R, D, L = (D, L, U)) & \text{gathering right part} \end{array}$$

After these rules, the contents of the R part is for (1) (D, U, R) which contains a copy of (SE, N, NE) and for the L part, (S, SW, NW). After that, the part C has all the necessary information to simulate one transition step of h (Fig. 9). The factor of deceleration is 2 as stated in Lemma 3.2. ∎

Below, we also recall Lemma 3.4 which states that a 6 neighbors PGA can be simulated by a 3 neighbors PGA (and conversely). It was proved by Róka by using Cayley graphs. But, since it is a local transformation, it will be used later.

Lemma 3.4 (Róka [13])**.** *N_6-PGA $\overset{1}{\prec}$ N_3-PGA (and conversely).* ∎

3.3. Homogeneous GAs and merging

Merging, the converse operation of splitting is more difficult. It is due to the fact that we need some special property of the GA for merging. Actually, merging is only possible if one of the graphs can be obtained by splitting from the other one (finding if a graph can be obtained by splitting seems to be a complex NP problem). But what is the acceleration factor?

Lemma 3.5. *Any GA $\mathcal{N}$ can be simulated by any other GA $m_G(\mathcal{N})$ obtained by application of a merge. The factor of acceleration equals one: in the worst case, there is no speedup.*

Proof. As we consider only radius 1 neighborhoods, the k-neighborhood of a node v is the set of all vertices at distance at most k from v. To simulate k steps of computation, an automaton needs to know the states of all vertices of the k-neighborhood of the simulated vertex.

Fig. 10 shows how such pieces of information about k-neighborhood can be difficult to gather in arbitrary networks. The subgraph on the left is G, our merging pattern. When we apply the merge on the large GA, we obtain a four vertices GA. But one can remark that the 2-neighborhood of the circled vertex is not contained in the 1-neighborhood of the macro-vertex in the resulting GA. Hence, we cannot really apply a simple speed-ud, as we need two steps to gather the 2-neighborhood of all vertices in each subgraph G. This is due to the size 4 of the grey face. On the right, there is a n-face, where the speed-up will be even more difficult. In arbitrary networks, arbitrary large faces occur. Hence there are arbitrary long paths that are not shrinked by the merge, preventing any acceleration (due to the "speed of light" limit inherently present in any automata network). Obviously, in some finite case, we can use more complex acceleration techniques, but the worst case can possibly occur.

■

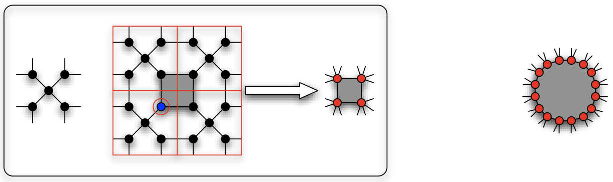


Figure 10: Acceleration problem.

However, many networks can be simulated with a good speedup factor by merging vertices. Hence, we must remember that we use some regular property of those networks and not only merging to obtain an acceleration. The theorem 3.7 is a refinement of the lemma 3.5. We obtain a more precise bound using the internal path length.

Definition 3.6. The *internal path length* of a subgraph with outgoing edges is the shortest path between two different outgoing edges. If there is a vertex with two outgoing edges, the *internal path length* is zero.

Theorem 3.7. *Any GA $\mathcal{N}$ can be simulated by any other GA $m_G(\mathcal{N})$ obtained by application of a merge. The factor of acceleration is at least one plus the internal path length of the subgraph G.*

Proof. Let i be the internal path length of G. In $m_G(\mathcal{N})$, the neighborhood of any vertex v contains the $(1+i)$-neighborhood of any vertex of $\mathcal{N}$ simulated by v. This is due to the fact that any path of length $1+i$ cannot reach outgoing edges of neighborhood subgraph G. ∎

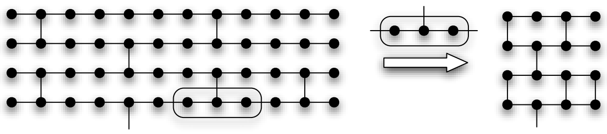


Figure 11: An efficient merging.

The figure 11 shows a merge with a subgraph G with an internal path length of 1.

3.4. Some efficient merges

Theorem 3.7 only gives a lower bound. We show now several ways to use regularity to find efficient merges to speed-up the simulations.

3.4.1. **n**-*ary tree.* In a infinite regular n-ary tree T_n, one can merge using any finite tree t. We consider oriented trees, each vertex having one father, and n sons.

The acceleration factor only depends upon the smallest path from the root of t to a descendant not in t. Hence we only consider complete n-ary tree of height h as t, that we note t_n^h.

$m_{t_n^h}(T_n) = T_{n^{h+1}}$, and the k-neighborhood in the merged tree contains the simulted $(1+(k-1)h)$-neighborhood. Hence, for any $\varepsilon > 0$, we can simulate T_n by $T_{n^{h+1}}$, with an acceleration factor of $h-\varepsilon$. Each automaton reads its k-neighborhood (with $k \geq \frac{h-1}{\varepsilon}$) using k time steps, and simulates $1+(k-1)h$ time steps of T_n. The factor of acceleration is $\frac{1+(k-1)h}{k} = h - \frac{h-1}{k} \geq h - \varepsilon$.

3.4.2. *Memory usage.* Hence, when one wants to simulate T_n at speed s, one can choose any $h > s$ to merge using t_n^h, and then $k = \lceil \frac{h-1}{h-s} \rceil$ will be the size of the neighborhood in the new network that we read before simulating several steps of computation. But which is the size of the simulating automaton? t_h^n contains $\frac{n^{h+1}-1}{n-1}$ vertices. And the k-neighborhood of a vertex in T_n contains $1+(n+1)\frac{n^k-1}{n-1}$ vertices (only 1 vertex if $k=0$). Let s be the speed we want to obtain, and h the height of t_n^h that we used for merging, the new automaton must contain at least:

$$m = \frac{n^{h+1}-1}{n-1}\left(1+(n^{h+1}+1)\frac{n^{(h+1)(\lceil \frac{h-1}{h-s} \rceil - 1)}-1}{n^{h+1}-1}\right)$$

copies of simulated automata. It means that a simulating automaton must remember the states of m different simulated automata to compute the simulation. By minimizing this formula, we obtain the best height to achieve the simulation at given speed with a minimum memory size for the new automaton. The minimal height h does not depend on n, and the height h that minimizes the memory size is obtained for k-neighborhood $k=2$ then $h = 2s-1$. This is the best way to simulate merged infinite regular trees. In this case, memory

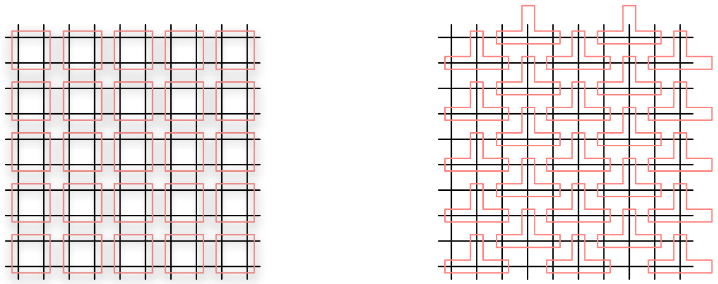


Figure 12: Simple (left) and optimal (right) merges of the grid for 4-vertices patterns.

usage grows as n^{4s}, which is quite fast. Hence merging is better than waiting (intuitively, merging allows several vertices to be merged and they can share memory, whereas waiting needs to copy on each vertex redondant pieces of information).

3.4.3. *Grids and toroidal meshes.* Merging grids is more complex, as any tiling pattern can be used to merge a grid, and it is an open question to know if a pattern tiles a grid [1]. Some tilings are surprisingly more efficient than others. Consider Fig. 12; on the left, we merge with squares, and as for trees, we can achieve any speed $2-\varepsilon$ (ε is expensive to minimize as in trees). On the right we merge using a pattern whose shape resembles a bottom symbol, and we achieve a speed of 2 with $k = 2$ which is optimal for a pattern of 4 vertices.

The results on trees hold on grids: merging is better than waiting. For example, when merging grids using squares, it is better to use big squares and read the 2-neighborhood than using smaller squares and read a larger neighborhood. The optimal memory usage (for a simulation at speed s) grows as $4s$.

4. Application to the simulations of GA on surfaces

We apply the neighborhood transformations introduced in section 3 to PGA embedded on surfaces. The simulations we construct have tight relations with some results on cellular automata on Cayley graphs recalled in the next section.

4.1. Related results

The results we present here come from the Cayley graph approach. Except for the definition of a local transition function (which is equivalent), they describe exactly the same objects but from a more algebraic point of view. We give here a statement of the results for finite PGAs embedded on surfaces as defined in Section 2.1.

Theorem 4.1 (Róka [11]). *$S_3N_6(r,p,q) \overset{1}{\prec} \overrightarrow{S_1}N_4(m,n)$ with $m = |\alpha_1(p+r-1) - \beta_1 p|$, $n = |\alpha_2(p-1) - \beta_2(p+q-1)|$, $(p-1)\alpha_1 = lcm(p-1, p+q-1)$, $(p+r-1)\alpha_2 = lcm(p+r-1, p)$, $(p+q-1)\beta_1 = lcm(p-1, p+q-1)$ and $p\beta_2 = lcm(p+r-1, p)$.* ■

Theorem 4.2 (Martin [5]). *$\overrightarrow{S_1}N_4(m,n) \overset{3.\min\{m,n\}+O(1)}{\prec} CylN_2(mn)$.* ■

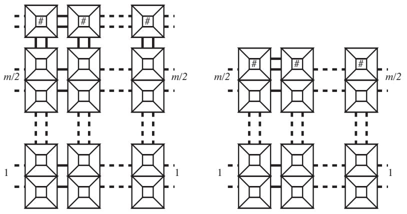


Figure 13: Simulating a PGA on a torus by a PGA on a cylinder (even and odd m).

And, by combining Theorem 4.1 and Theorem 4.2, we get:

Corollary 4.3. *$S_3N_6(r,p,q) \overset{3.\min\{m,n\}+O(1)}{\prec} CylN_2(mn)$ where m and n are those of Theorem 4.1.* ■

Theorem 4.4 (Martin-Peyrat [7])**.** *$\overrightarrow{S_1}N_4(m,n) \overset{\Theta(n+m)}{\prec} CylN_2(mn)$ if and only if $n \equiv 2$ mod m; in this case the number of copies of each cell is minimal.* ■

Theorem 4.4 improves the time-complexity of Theorem 4.2. We have minimized the number of copies requested to simulate the behavior of a torus of $n \times m$ automata by a cylinder of $n \cdot m$ automata. Observe that this number of copies cannot be further improved. It is thus the minimal number of copies requested to complete this task. Theorem 4.4 forbids some values of n and m. However, for those prohibited values, one can use Theorem 4.2.

4.2. New simulation results

Below, we propose a series of finite simulation results. The first (Lemma 4.5) proposes a simulation of a N_4-PGA embedded on a torus by a N_4-PGA embedded on a cylinder. To do this, we need to cut the torus along one of its Jordan curves.

Lemma 4.5. *$\overrightarrow{S}_1N_4(n,m) \overset{1}{\prec} CylN_4(n,\lceil \frac{m}{2} \rceil)$.*

(Proof sketch). We consider a N_4-PGA with (m,n) nodes embedded on the torus with m beeing the height and n the width. We "cut" all the connections along the width and we fold the resulting cells on the middle. This construction is analogous with the simulation of a Turing machine with a bi-infinite tape by a Turing machine with an (simply) infinite tape. We add a "dummy" cell (with symbol $\sharp$) on the top depending upon the parity of m (see Fig. 13). It is not difficult to design the local transition function of the new PGA. ■

Theorem 4.6 proposes two simulations of a N_6-PGA embedded on a torus. The first one by a N_4-PGA embedded on a torus and the other by a N_4-PGA embedded on a cylinder. The results are easily obtained from the previous lemmas.

Theorem 4.6. *$\overrightarrow{S}_1N_6(n,m) \overset{2}{\prec} \overrightarrow{S}_1N_4(2n,m)$, and $\overrightarrow{S}_1N_6(n,m) \overset{2}{\prec} CylN_4(n,m)$.* ■

Conclusion

This paper proposed two graph transformations: splitting and merging. Both can be used for simulating graph automata in order to make local transformations on the neighborhood. When used on regular graphs, the results we obtain can be compared with those for cellular automata on Cayley graphs. In particular, they can be used in this formalism and, conversely, local transformations on Cayley graphs can be applied to graph automata. We have given some examples of this kind in the paper. Only Lemma 4.5 cannot be related to cellular automata on Cayley graph since the PGA on a cylinder we have built for the simulation is not regular. The results of this paper can be combined with results from the Cayley graph approach to give other results as, for instance, one can replace N_6 by N_3 in Theorem 4.6. This study of the relationships –and differences– between both approaches would be interesting to pursue.

References

[1] D. Beauquier and M. Nivat. Tiling the plane with one tile. In *Symposium on Computational Geometry*, pages 128–138, 1990.

[2] C. Berge. *Graphes.* Gauthier Villars, third edition, 1983.

[3] M. Henle. *A combinatorial introduction to topology.* Dover Publication, 1979.

[4] K. Kuratowski. *Introduction à la théorie des ensembles et à la topologie.* Institut de Mathématiques de l'Université de Genève, 1966.

[5] B. Martin. Embedding torus automata into a ring of automata. *Int. Journal of Found. of Comput. Sc.*, 8(4):425–431, 1997.

[6] B. Martin. A simulation of cellular automata on hexagons by cellular automata on rings. *Theoretical Computer Science*, 265:231–234, 2001.

[7] B. Martin and C. Peyrat. A single-copy minimal-time simulation of a torus of automata by a ring of automata. *Discrete Applied Math.*, 155:2130–2139, 2007.

[8] K. Morita and M. Harao. Computation universality of one-dimensional reversible (injective) cellular automata. *Trans. IEICE Japan*, E(72):758–762, 1989.

[9] K. Morita, M. Margenstern, and K. Imai. Universality of reversible hexagonal cellular automata. In *Workshop on Frontiers between Decidability and Undecidability*, Brno, 1998.

[10] Zs. Róka. One–way cellular automata on Cayley graphs. In *Proc. FCT'93*, number 710 in Lecture Notes in Computer Science, pages 406–417. Springer Verlag, 1993.

[11] Zs. Róka. *Automates cellulaires sur graphes de Cayley.* PhD thesis, École Normale Supérieure de Lyon, 1994.

[12] Zs. Róka. One–way cellular automata on Cayley graphs. *Theoretical Computer Science*, 132(1–2):259–290, 1994.

[13] Zs. Róka. Simulations between cellular automata on Cayley graphs. *Theoretical Computer Science*, 225:81–111, 1999.

Journées Automates Cellulaires 2008 (Uzès), pp. 102-118

UNIVERSALITIES IN CELLULAR AUTOMATA A (SHORT) SURVEY

NICOLAS OLLINGER [1]

[1] Laboratoire d'informatique fondamentale de Marseille (LIF)
Aix-Marseille Université, CNRS,
39 rue Joliot-Curie, 13 013 Marseille, France
E-mail address: `Nicolas.Ollinger@lif.univ-mrs.fr`

Abstract. This reading guide aims to provide the reader with an easy access to the study of universality in the field of cellular automata. To fulfill this goal, the approach taken here is organized in three parts: a detailled chronology of seminal papers, a discussion of the definition and main properties of universal cellular automata, and a broad bibliography.

Introduction

The idea and construction of a universal cellular automaton is as old as the formal study of the object itself, starting with the work of von Neumann [82] on self-reproduction in the 1940s, using cellular automata under suggestions by Ulam. Following the work of Turing, a Turing-universal cellular automaton is an automaton encompassing the whole computational power of the class of Turing machines, or by so-called Church-Turing thesis the class of recursive functions. To encode complex behaviors in a cellular automaton's dynamics, one can describe how to encode any computing device of a universal class of machines (Turing machine, tag systems, etc) and use classical tools of computability theory to shape wanted behaviors of the object. This is basically what von Neumann did. He designed a cellular automaton able to encode any Turing machine, the machine being moreover equipped with a construction arm controlled by the machine's head.

But Turing-universality is not the only reasonable kind of universality one might expect from cellular automata. It is quite unnatural to consider a universality of highly parallel potentially infinite devices as cellular automata by simulation of the dynamics of sequential finite machines — indeed, as we will discuss, to give a both widely acceptable yet precise definition of Turing-universality is a very difficult and unfulfilled challenge. As the study of cellular automata shifted both to dimension 1 and to the study of its dynamics, another kind of universality emerged. An intrinsically universal cellular automaton is an automaton able to properly simulate the behavior of any other cellular automaton on any type of

2000 ACM Subject Classification: F.1.1.
Key words and phrases: cellular automata, universality, reversibility.
This research has been supported by the French national research agency (ANR) grant Sycomore.

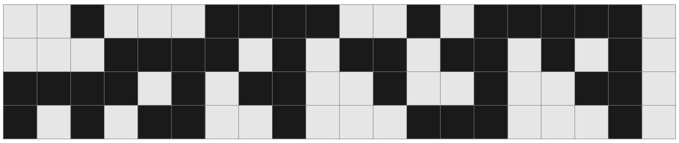


Figure 1: Partial space-time diagram of the $(\mathbb{Z}_2, +)$ rule

configuration (might it be infinite). It turns out that most of the historical constructions in dimension 2 and more, whereas designed as Turing-universal are Intrinsically Universal by the simple fact that they are designed to encode any boolean circuit.

A formal definition of universality might not seem so important. In fact, when building a precise cellular automaton from scratch to be universal, a definition is often implicit: the obtained behavior is the one engineered by the designer. The definition turns out to be more required when proceeding by analysis: given a cellular automaton rule, is it universal?

The present reading guide is constructed as follows. Section 1 *Cellular Automata* (p. 103) gives the definitions and notations used for cellular automata, configurations, and dynamics. Section 2 *Chronology* (p. 105) is an annotated chronology of seminal papers preparing to and concerning universality and universal cellular automata. Section 3 *Towards Formal Definitions* (p. 108) discusses the right definition of universalities in cellular automata. Section 4 *Higher Dimensions* (p. 110) discusses the construction and analysis of universal cellular automata in dimensions 2 and more, mostly using boolean circuits simulation. Section 5 *Turing Universality* (p. 111) discusses Turing universality, its links with universal Turing machines and the main technics of construction. Section 6 *Intrinsic Universality* (p. 113) discusses Intrinsic universality and the main technics of construction. Section 7 *Reversiblity and Universality* (p. 115) discusses universality in the special restricted case of reversible cellular automata.

1. Cellular Automata

A *cellular automaton* $\mathcal{A}$ is a tuple (d, S, N, f) where d is the *dimension* of space, S is a finite set of *states*, N a finite subset of $\mathbb{Z}^d$ is the *neighborhood* and $f : S^N \to S$ is the *local rule*, or *transition function*, of the automaton. A *configuration* of a cellular automaton is a coloring of the space by S, an element of $S^{\mathbb{Z}^d}$. The *global rule* $G : S^{\mathbb{Z}^d} \to S^{\mathbb{Z}^d}$ of a cellular automaton maps a configuration $c \in S^{\mathbb{Z}^d}$ to the configuration $G(c)$ obtained by applying f uniformly in each cell: for all position $z \in \mathbb{Z}^d$, $G(c)(z) = f(c(z + \nu_1), \ldots, c(z + \nu_k))$ where $N = \{\nu_1, \ldots, \nu_k\}$. A *space-time diagram* of a given cellular automaton is a mapping $\Delta \in S^{\mathbb{N}\times\mathbb{Z}^d}$ such that for all time step $t \in \mathbb{N}$, $\Delta(t + 1) = G(\Delta(t))$.

Example 1.1. Fig. 1 is a partial representation of a space-time diagram of the cellular automaton $(1, \mathbb{Z}_2, \{0, 1\}, f)$ where $f(x, y) = x + y$. State 0 is represented by the white color, state 1 by the black color. Time goes from bottom to top.

In this paper, we consider for the most part cellular automata of dimension 1 and 2 with the typical neighborhoods depicted on Fig. 2: von Neumann $\{(-1, 0), (1, 0), (0, -1), (0, 1)\}$ and Moore $\{-1, 0, 1\}^2$ in dimension 2, first neighbors $\{-1, 0, 1\}$ and one way $\{-1, 0\}$ in dimension 1.

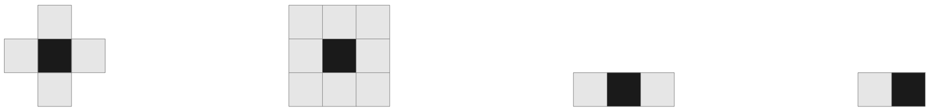


Figure 2: Typical neighborhoods

Several subsets of the space of configurations are considered. Given a *quiescent state* q satisfying $f(q,\dots,q)=q$, a *q-finite configuration* c is a configuration equal to q in all but finitely many cells: there exists α such that for all position $z\in\mathbb{Z}^d$, $\|z\|_\infty>\alpha\rightarrow c(z)=q$. A configuration c admits p as a *periodicity vector* if for all position $z\in\mathbb{Z}^d$, $c(z+p)=c(z)$. A configuration c in dimension d is *periodic* if it admits a family of d non-colinear periodicity vectors: there exists $p\in\mathbb{N}^d$ such that $(p_1,0,\dots,0)$, $(0,p_2,0,\dots,0)$, $\dots$, and $(0,\dots,0,p_d)$ are periodicity vectors of c. A configuration c in dimension d is *ultimately periodic* if there exists α and d non-colinear vectors v_i such that for all position $z\in\mathbb{Z}^d$ and all vector v_i, $\|z\|_\infty>\alpha\rightarrow c(z+v_i)=c(z)$. Notice that in dimension 1, an ultimately periodic configuration can have two different ultimately periodic pattern on each side.

Constraints can also be added to the local rule. Symmetries are usually considered to obtain more natural rules mimicking physical systems. A symmetry rule can be seen as a one-to-one mapping $\rho:\mathbb{Z}^d\rightarrow\mathbb{Z}^d$: the image of a configuration $c\in S^{\mathbb{Z}^d}$ by the symmetry rule ρ is the configuration $\rho(c)$ satisfying for all position $z\in\mathbb{Z}^d$, $\rho(c)(z)=c(\rho(z))$. A cellular automaton $\mathcal{A}$ respects a symmetry rule ρ if ρ and G commute, *i.e.* $\rho(G(c))=G(\rho(c))$. Typical symmetries include reflections around point $(\rho_0(x,y)=(-x,-y))$, around axes $(\rho_x(x,y)=(-x,y))$ and rotations $(\theta(x,y)=(-y,x))$. A cellular automaton is *totalistic* if it's set of states is a subset of $\mathbb{N}$ and the local rule f can be written as $f(s_1,\dots,s_k)=g(\sum_{i=1}^k s_i)$. Totalistic rules respect all symmetries that preserve the neighborhood (*i.e.* such that the image of the neighborhood by the symmetry rule is equal to the neighborhood): totalistic cellular automata with the von Neumann or Moore neighborhood are reflection and rotation invariants.

A cellular automaton is injective (resp. surjective, one-to-one) if its global rule is injective (resp. surjective, one-to-one). A cellular automaton $\mathcal{A}$ is reversible if there exists a cellular automaton $\mathcal{B}$ that reverts it, that is such that $G_\mathcal{B}\circ G_\mathcal{A}$ is the identity map.

Proposition 1.2 (Hedlund [31], Richardson [72])**.** *A cellular automaton is reversible if and only if it is injective.*

Proposition 1.3 (Amoroso and Patt [2])**.** *It is decidable given a one-dimensional cellular automaton to decide whether it is reversible.*

Proposition 1.4 (Kari [34, 35])**.** *It is undecidable given a two-dimensional cellular automaton to decide whether it is reversible.*

Whereas reversibility is an undecidable question, the construction of reversible cellular automata is possible, provided that the backward rule is constructed at the same time as the forward rule. Partitioned cellular automata provide a convenient way to construct reversible cellular automata. A *partitioned cellular automaton* is a cellular automaton with state set

$S_1 \times S_2 \times \cdots \times S_k$ whose local rule can be rewritten as $f((s_1^1, \ldots, s_1^k), \ldots, (s_k^1, \ldots, s_k^k)) = \varphi(s_1^1, s_2^2, \ldots, s_k^k)$ where $\varphi : \prod S_i \to \prod S_i$ is the partitioned rule. As it is straightforward to verify, a partitioned cellular automaton is reversible if and only if its partitioned rule is one-to-one. As the partitioned rule is a mapping from a finite set to itself, any partially defined injective rule can be completed to a reversible cellular automaton.

For a better and more complete introduction to the theory of cellular automata, see Delorme [18] and/or Kari [36].

2. Chronology

It is a difficult task to give a fair and complete chronology of a research topic. In this section, we propose an exploration of the history of the field in three main eras:

(1) the *computation and machines* era describes seminal papers outside the realm of cellular automata that lead to the main tools necessary to consider computation in the context of abstract machines;
(2) the *universality and cellular automata* era is the core part of the chronology: it describes seminal papers along the path of universality study in the realm of cellular automata, from the early work of von Neumann in the 50s to the end of the 90s;
(3) the *recent trends* era is a more subjective choice of some papers in the field in the twenty-first century.

2.1. Computation and Machines

Gödel 1931 [30]**:** in his now classical paper describing incompleteness theorems, Gödel introduces so-called Gödel numberings: the ability to encode and manipulate a formal system inside itself if the system is complex enough. The concept of universality directly depends on such an encoding: a universal machine simulates a machine through its encoding. For a precise analysis from a logic and computer science point of view of Gödel's paper, see Lafitte [38].

Turing 1936 [81]**:** while introducing Turing machines and proving the undecidability of the halting problem by a diagonal argument, Turing also introduces its universal machine. Fixing an enumeration of Turing machines and a recursive bijective pairing function $\langle ., . \rangle : \mathbb{N}^2 \to \mathbb{N}$, he describes a machine U that, on input $\langle m, n \rangle$, computes the same value as the machine encoded m on input n. Universality as a property is not discussed: a unique universal Turing machine U is given. For a discussion of the development of ideas from Leibniz to Turing results, see Davis [17].

Post 1943 [68]**:** at that time, many different models of computation where proposed and proved equivalent, leading to the so-called Church-Turing thesis. Post introduces tag systems, a combinatorial word-based system successfully used since to construct size-efficient universal Turing machines. For a modern definition and discussion of tag systems, see Minsky [51].

Kleene 1956 [37]**:** finite state machines are at the hearth of many models of computation. Kleene's paper proves the equivalence between three different families of objects: regular languages, finite automata, and boolean circuits. Boolean circuits are modeled after the formal study of abstract neurons by McCulloch and Pitts [49]. This equivalence is fundamental both to concrete computer design and discrete models of computation like cellular automata. For a modern discussion on this equivalence and its consequences from the point of view of computation and machines, see Minsky [51]. Perrin [66] gives an history of this period and important achievements with respect to the field of finite automata and formal languages.

Minsky 1967 [51]**:** In the spirit of the question from Shannon [73] about the size of a smallest Turing machine, Minsky explains how to efficiently encode tag systems computations into Turing machines and describe a universal Turing machine with four symbols and seven states. This marks the real start of a (still running) competition.

Lecerf 1963 [40]**, Bennett 1973** [8]**:** Reversible computation is concerned with computing devices that can unroll their computation, going back in time. In their independent papers, Lecerf and Bennett prove that reversible Turing machines are able to simulate just any Turing machine. Thus, there exists reversible Universal Turing machines.

Fredkin and Toffoli 1982 [27]**:** To encode classical computations into discrete models, Kleene [37]'s theorem permits to go freely from circuits to finite state machine, an essential ingredient for computation. Fredkin and Toffoli discuss an analogous for reversible computation: elementary building blocks to encode any reversible finite state machine as a reversible circuit. This paper also introduces the so-called billard ball model of computation: a discrete cellular automata model to encode reversible computations. The encoding of reversible finite state machines into circuits was later improved by Morita [55].

2.2. Universality and Cellular Automata

von Neumann 1966 [82]**:** Introducing cellular automata in order to construct a self-reproducing machine, participating to the reflexion on the nature of life, von Neumann takes a fixed-point approach. His two-dimensional, 29 states, von Neumann neighborhood cellular automaton is able to simulate a particular kind of Turing machine that can also control a construction arm. The power of the construction arm is rich enough to construct with finitely many instructions a copy of the Turing machine itself. Whereas the machine is constructed with a form of Turing-universality in mind, the simulation of the Turing machine is done with very simple components wiring down a particular family of boolean circuits. As a consequence, the original cellular automaton is also, *a posteriori*, intrinsically universal. The construction of von Neumann leads to various improvements and discussions on the encoding of boolean circuits, the different organs that compose the machine and the transmission of signals. A non exhaustive list of interesting following papers might be: Arbib [3], Burks [10, 11, 12], Moore [52, 53], Thatcher [77, 76].

Codd 1968 [14]**:** Following the principle of von Neumann idea on self-reproduction, Codd drastically reduces the complexity of the automaton. Codd's two-dimensional rule uses 8 states with the von Neumann neighboorhood. Signals are conveyed by pairs of states (an oriented particle) moving between walls and reacting upon collision. This cellular automaton is also universal for boolean circuits and so intrinsically universal. A latter construction by Langton [39], based on Codd ideas, has fewer states and a very simple family of self-reproducing loops but looses its computation universal capabilities.

Banks 1970 [4, 5]**:** The work of Banks is noticeable with respect to several aspects and also because of its relatively small diffusion in the cellular automata community. Banks constructs a family of very small cellular automata (two-dimensional, von Neumann neighborhood, very symmetric, four to two states) simulating boolean circuits in a very simple and modern way (signals moving in wires, boolean gates on collisions), he identified and used explicitly the property of intrinsic universality and gave a transformation to construct relatively small universal one-dimensional cellular automata with large neighborhoods starting from two-dimensional ones (reencoding it into a one-dimensional first-neighbors automaton with 18 states). Construction of a two-dimensional four state universal cellular automaton in the spirit of Banks is provided by Noural and Kashef [61].

Conway 1970 [29, 9]**:** The Game of Life introduced by Conway is certainly among the most famous cellular automata and the first rule to be proven universal by analysis of a given rule rather than on purpose construction. A modern exposition of the Game of Life universality and a proof of its intrinsic universality was latter proposed by Durand and Róka [22].

Smith III 1971 [74]**:** The simulation of Turing machine by cellular automata to construct one-dimensional Turing-universal cellular automata is studied by Smith III. Among several results, he explains how to construct a one-dimensional Turing-universal cellular automaton with first neighbors and 18 states.

Toffoli 1977 [80]**:** Any cellular automaton of dimension d can be simulated, in a certain sense, by a cellular automaton of dimension $d+1$. Using this assertion, Toffoli shows that two-dimensional reversible cellular automata can be Turing-universal. The result was later improved by Hertling [32].

Margolus 1984 [42]**:** Whereas Toffoli transforms any Turing machine into a two-dimensional cellular automaton by using a new spatial dimension to store computational choices, Margolus constructs a Turing-universal two-dimensional reversible cellular automaton by simulation a bouncing billard ball, complex enough to compute any reversible boolean function of conservative logic. The billard ball model cellular automaton has 16 states defined as two-by-two blocks of binary cells and von Neumann neighborhood.

Albert and Čulik 1987 [1]**:** Each cellular automaton can be simulated by a totalistic cellular automaton with one-way neigborhood. With the help of the last proposition, Albert and Čulik construct the first universal cellular automaton obtained by simulation of any cellular automaton of the same dimension. The automaton works along the following principle: each macro-cell copies the state of its left neighbor and adds it to its state obtaining some n, then by copying the nth element of a reference table, it selects its new state. Whereas the spirit of intrinsic universality is definitely there, the technical implementation is less clear. The one-dimensional first-neighbors automaton obtained has 14 states. The construction was later improved by Martin [43, 44] with better transition time complexity and smn theorem.

Morita and Harao 1989 [57]**:** Introducing partitioned cellular automata, Morita and Harao explicitely simulate any reversible Turing machine on a one-dimensional reversible cellular automaton, proving that one-dimensional reversible cellular automata can be Turing-universal. The construction was later improved by Dubacq [21], simulating any Turing machine in real time (without loss of time).

Lindgren and Nordahl 1990 [41]**:** The direct simulation of Turing machine on one-dimensional cellular automata proposed by Smith III can be improved and any m states n symbols machine can be simulated by a $(m + n + 2)$-states cellular automaton following Lindgren and Nordhal. Applying this to Minsky's 7 states and 4 symbols machine and then transforming the simple simulation into a macro-state signal based simulation, Lindgren and Nordhal obtain a one-dimensional first neighbors 7 state Turing-universal cellular automaton. The intrinsic universality status of this automaton is unknown.

Durand and Róka 1996 [22]**:** Revisiting the Game of Life and filling holes in the universality proof, Durand and Róka publish the first discussion on the different kinds of universality for cellular automata and the problem of formal definition.

Durand-Lose 1997 [25]**:** Using a modern definition of intrinsic universality, Durand-Lose goes one step further than Morita and Harao by constructing a one-dimensional cellular automata intrinsically simulating any cellular automaton.

2.3. Recent Trends

Imai and Morita 2000 [33]**:** The improvement in the construction of small and simple two-dimensional reversible cellular automata continues. Imai and Morita use partitioned cellular automata to define an 8 state universal automaton.

Ollinger 2002 [64]**:** Using simulation technics between cellular automata, strong intrinsically universal cellular automata with few states can be constructed, here 6 states.

Cook 2004 [15]**:** Very small universal cellular automata cannot be constructed, they have to be obtained by analysis. Realising a real tour de force, Cook was able to prove the Turing-universality of the 2-states first-neighbors so-called rule 110 by analysing signals generated by the rule and their collisions. The intrinsic universality of this automaton remains open. The original construction, simulation of a variant of tag system, was exponentially slow. For a proof of Cook's result using signals, see Richard [70].

Neary and Woods 2006 [60]**:** Recently, the prediction problem of rule 110 was proven P-complete by Neary and Woods by a careful analysis and modification of Turing machines simulation technics by tag systems. As P-completeness is required for intrinsic universality, this is another hint of the potential strong universality of rule 110.

Richard 2008 [71]**:** The limits of constructed small intrinsically universal cellular automata are converging towards analysed cellular automata. Using particles and collisions, Richard was recently able to construct a 4 state intrinsically universal first-neighbors one-dimensional cellular automaton.

3. Towards Formal Definitions

What is a universal cellular automaton? At first, the question might seem simple and superficial: a universal cellular automaton is an automaton able to compute anything recursive. In fact, a formal definition is both required and difficult to obtain. The requirement for such a definition is needed to define a frontier between cellular automata of maximal complexity and the others: in particular when considering the simplest cellular automata, to be able to identify the most complex. The difficulty arise from the fact that we both want a definition broad enough to encapsulate all constructions of the literature and all *fair enough* future constructions. For more details concerning this philosophical question, see Durand and Róka [22] attempt to give formal definitions.

Turing-universality is the easiest form of universality one might think about, that is with a computability culture: let the cellular automaton *simulate* a well known universal model of computation, either simulating one universal object of the family or any object of the family.

The first approach pushes back the problem to the following one: what is a universal Turing machine? a universal tag system? In its original work, Turing did not define universal machines but *a* unique universal machine. The definition of universality for Turing machines was later discussed by Davis [16] who proposed to rely on recursive degrees, defining universal machines as machines with maximal recursive degree. This definition while formal lacks precise practical view of encoding problems: the issue continues to be discussed in the world of Turing machines, becoming more important as smaller and smaller universal machines are proposed. For a view on the universality of Turing machines and pointers to literature related to the topic, see Woods [86].

The second approach leads to the problem of heterogeneous simulation: classical models of computation have inputs, step function, halting condition and output. Cellular automata have no halting condition and no output. As pointed out by Durand and Róka [22], this leads to very tricky encoding problems: their own attempt of a Turing-universality based on this criterium as encoding flaw permitting counter-intuitively to consider very simple cellular automata as universal.

Turing-universality of dynamical systems in general and cellular automata in particular has been further discussed by Delvenne, Kůrka, and Blondel [19], and Sutner [75]. None of the proposed definition are completely convincing so forth, so we will choose on purpose not to provide the reader with yet another weak formal definition.

Intrinsic universality, on the other hand, is easier to formalize, yet more robust notion (in the sense that variations along the lines of the definition lead to the same set of universal automata). Consider a homogenous type of simulation: cellular automata simulated by cellular automata in a shift invariant, time invariant way. A natural type of universal object exist in this context: cellular automata able to simulate each cellular automaton. Following the ideas of grouping and bulking [69, 47, 63], we introduce a general notion of simulation broad enough to scope all reasonable constructions of the literature.

Direct simulation between two cellular automata can be formalized as follows. A cellular automaton $\mathcal{B}$ *directly simulates* a cellular automaton $\mathcal{A}$, denoted $G_{\mathcal{A}} \prec G_{\mathcal{B}}$, of the same dimension according to a mapping $\varphi : S_{\mathcal{A}} \to 2^{S_{\mathcal{B}}}$ if for any pair of states $a, b \in S_{\mathcal{A}}$, $\varphi(a) \cap \varphi(b) = \emptyset$ and for any configuration $c \in S_{\mathcal{A}}^{\mathbb{Z}^d}$, $G_{\mathcal{B}}(\varphi(c)) \subseteq \varphi(G_{\mathcal{A}}(c))$.

For any state set S, let $(m_1, \ldots, m_d)$ be a tuple positive integers, the *unpacking bijective map* $o_{(m_1,\ldots,m_d)} : \left(S^{\prod m_i}\right)^{\mathbb{Z}^d} \to S^{\mathbb{Z}^d}$ is defined for any configuration $c \in \left(S^{\prod m_i}\right)^{\mathbb{Z}^d}$ and any position $z \in \mathbb{Z}^d$ and $r \in \prod_i \mathbb{Z}_{m_i}$ as $o_{(m_1,\ldots,m_d)}(c)(m_1 z_1 + r_1, \ldots, m_d z_d + r_d) = c(z)(r)$. The *translation* of vector $v \in \mathbb{Z}^d$ is defined for any configuration $c \in S^{\mathbb{Z}^d}$ and position $z \in \mathbb{Z}^d$ as $\sigma_v(c)(z) = c(z - v)$.

Simulation between two cellular automata is extended by considering packing, cutting and shifting of the two cellular automata such that direct simulation occur between both transformed objects. Universal objects are then maximum of the induced pre-order. In fact, it can be proved that simulation on one side is sufficient for universal objects.

Definition 3.1 (intrinsic universality)**.** A cellular automaton $\mathcal{U}$ is intrinsically universal if for each cellular automaton $\mathcal{A}$ of the same dimension there exists an unpacking map o_m, a positive integer $n \in \mathbb{N}$ and a translation vector $v \in \mathbb{Z}^d$ such that $G_{\mathcal{A}} \prec o_m^{-1} \circ G_{\mathcal{U}}^n \circ o_m \circ \sigma_v$.

Proposition 3.2 (Mazoyer and Rapaport [69, 47])**.** *No cellular automaton is intrinsically universal in real time (that is, when constraining cutting constant n to be equal to* $\max(m)$*): simulation cannot perform both information displacement and transition computation at the same time.*

Proposition 3.3 (Ollinger [65])**.** *Given a cellular automaton, it is undecidable to determine whether it is intrinsically universal.*

Turing-universality and intrinsic universality notions are really different notions. Some erroneous claims by Wolfram [83, 84] affirm for example that rule 110 is intrinsically universal. In fact, the question is yet open, Turing universality is the only proven thing.

Proposition 3.4 (Ollinger [63], Theyssier [78])**.** *There exists Turing-universal cellular automata which are not intrinsically universal. Moreover, some of them are at the bottom of an infinite increasing chain of equivalences classes of the preorder.*

Universality can also be discussed when considering language recognition or computation on grids. This topic is out of scope of the present paper. For more on this topic, see Mazoyer [45, 46].

4. Higher Dimensions

In two and more dimensions, an easy way to construct both intrinsically and Turing universal cellular automata is to go through boolean circuit simulation. Boolean circuits can encode any finite state machine and a cell of a cellular automaton or the control and tape of a Turing machine can be described as finite state machines. The topic of boolean circuit simulation with cellular automata is quite popular and a lot has been written on it, see for example recreations around the wireworld cellular automaton designed by Silverman and discussed in Dewdney [20]. Let us just here give technical hints and possible exotic extensions without entering details.

To simulate boolean circuits, one typically needs to mix the following ingredients:

wires: boolean signals travel in piecewise straight line in space, their paths are the wires. Several encoding of boolean signals with or without explicit wires are possible: moving particles encoded as states with a direction vector, bouncing on walls to turn as in game of life [29]; wire path encoded as wire cells with explicit direction vector on each wire cell as in von Neumann [82]; undirected wire cells on which directed signals travel; undirected wire cells on which pairs of two different signal cells travel, the direction being given by the orientation of the pair as in wireworld [20]; pairs of undirected wire paths in between which pairs of two different signal cells travel as in Codd [14] or Banks [5].

turn and delay: boolean signals should be able to turn in space and delay their arrival to permit signal synchronization.

signal crossing: in order to encode all boolean circuits, crossing of signals has to be encoded either explicitly (adding crossing states) or implicitly using delaying technics (as in von Neumann [82]) or boolean logic tricks.

gates: signals must be combined using boolean gates at least taken in a boolean universal family of gates. AND, OR, NOT is the classical one but NAND or NOR is sufficient alone.

fan-out: signals must be duplicated in some way either with an explicit fan-out state or using specific wire split rules.

Remarks and encoding tricks regarding boolean circuit simulation:

- Universal boolean functions families and their expressive power are described in Post [67]. But, in cellular automata encoding, it is easy to use constants and multiple wires encoding, thus the number of boolean classes depending on the implemented gates is finite and small.
- Clocks are only needed when dealing with some form of synchronized logic simulation. It is often used because boolean signals are encoded with two values: empty wire or signal on wire. With such an encoding, NOT gate has to generate new signal on wire and clock signal is used to determine at which time steps to do so. However, a classical coding trick to avoid the use of clocks and diods is to only implement OR and AND gates and use the two wires trick to gain boolean universality: a signal is encoded as one signal on one wire, the second being empty (thus no signal is encoded as no signal on both wires), then the NOT gate is just the wire crossing $(x, y) \mapsto (y, x)$, the AND gate can be encoded as $(x, y) \mapsto (x \wedge y, x \vee y)$ and the OR gate as $(x, y) \mapsto (x \vee y, x \wedge y)$. As both OR and AND produce signal only if there is at least one signal in input, the need for clock vanishes.
- Wire crossing can be gained for free by using the XOR gate as planar crossing can be implemented with XORs.
- Delays come for free if the wires can turn in all directions.
- In dimension 3, wire crossing is not needed, use the third dimension to route wires.
- Signal encoding can be done using signal constructions, in the spirit of Mazoyer and Terrier [48], in order to reduce the number of states.

Of course, boolean circuit simulation is not restricted to square grids. As an example of a more exotic lattice, Gajardo and Goles [28] encoded a boolean circuit simulator on a hexagonal lattice (with proper cellular automata definition).

Small intrinsically universal cellular automata are quite simple to construct in dimension two with few states: Banks does it with 2 states and von Neumann neighborhood with reflection and rotation symmetry.

5. Turing Universality

In dimension one, boolean circuit encoding is more puzzling as wire crossing capabilities is bounded by the local rule. Thus, historically, computation universality is achieved by direct simulation of universal models of computations :

Turing machines: Turing machines are easy to encode on cellular automata (see below) as an infinite tape really looks like a configuration of a cellular automaton.

In fact, several variants of Turing machines exist and an important literature on universality in the Turing world provide useful objects to build small universal automaton based on this model. The question of existence of small universal Turing machines was first raised by Shannon [73], different variants of Turing machines are discussed by Fischer [26]. For a survey on small Turing machine construction, see Woods and Neary [86].

Tag systems: Tag systems provide a better model to design very small universal objects. In fact, very small universal Turing machines are constructed by simulation of tag systems and their variants as originally proposed by Minsky [51, 13]. The original drawback of tag system was its exponential slow-down when simulating Turing machines, this drawback was removed recently by Woods and Neary [85, 86] achieving polynomial time simulation. The Turing-universality of rule 110 is obtained by Cook [15] by direct simulation of a proper variant of tag systems.

The variant of Turing machine we use is the following. A *Turing machine* is a tuple (S, Σ, B, s_0, T) where S is a finite set of states, Σ is a finite alphabet with a special blank symbol $B \in \Sigma$, $s_0 \in S$ is the initial state and $T : S \times \Sigma \to S \times \Sigma \times \{\leftarrow, \rightarrow\}$ is a partial transition map. A transition rule $T(s, a) = (s', b, d)$ reads as follow: when reading a from state s, write b on the tape, move in direction d, and enter state s'. A configuration of the machine is a triple (s, z, c) where $s \in S$ is the current state of the machine, $z \in \mathbb{Z}$ is the position of the head, and $c \in S^{\mathbb{Z}}$ is the content of the tape. The machines goes in one step from a configuration (s, z, c) to a configuration (s', z', c') if the transition rule $T(s, c(z)) = (s'', d, b)$ is defined and verifies $s' = s''$, $z' - z = d$, $c'(z) = b$ and for all position $z'' \neq z$, $c'(z) = c(z)$. Starting from a configuration $\mathfrak{c}$, an halting computation of the machine in time t consists of a sequence of configurations $(\mathfrak{c}_i)_{i=0}^{t}$ such that $\mathfrak{c}_0 = \mathfrak{c}$, the machine cannot reach any configuration from $\mathfrak{c}_t$ and for all i, the machine goes in one step from $\mathfrak{c}_i$ to $\mathfrak{c}_{i+1}$. The configuration $\mathfrak{c}_t$ is the output of the computation.

Following Smith III [74], a given Turing machine (S, Σ, B, s_0, T) can be simulated by a cellular automaton $(1, S', \{-1, 0, 1\}, f)$ as follows. Let $S' = \Sigma \cup S \times \Sigma$. A configuration (s, z, c) of the Turing machine is encoded as a configuration $c' = \tau(s, z, c)$ of the cellular automaton in the following way: $c'(z) = (s, c(z))$ and for all positions $z' \neq z$, $c'(z') = c(z)$. The local rule encodes the transition function of the Turing machine. For each transition $T(s, a) = (s', b, \leftarrow)$, for all states $x, y \in S$, $f(x, y, (s, a)) = (s', y)$ and $f(x, (s, a), y) = b$. Symmetrically, for each transition $T(s, a) = (s', b, \rightarrow)$, for all states $x, y \in S$, $f((s, a), y, x) = (s', y)$ and $f(x, (s, a), y) = b$. All undefined transitions apply identity: $f(x, y, z) = y$. With this encoding, starting from an encoded configuration $\tau(\mathfrak{c})$, the configuration evolves in one step to a configuration $\tau(\mathfrak{c}')$ where $\mathfrak{c}'$ is the next computation step of the Turing machine if it exists, $\mathfrak{c}' = \mathfrak{c}$ otherwise. Using this simulation, a Turing machine with m states and n symbols is simulated by a one-dimensional cellular automaton with first-neighbors and $(m + 1)n$ states.

To lower the number of states, Lindgren and Nordahl [41] introduce a simulation scheme where each step of the Turing machine computation is emulated by two time steps in the cellular automaton. A given Turing machine (S, Σ, B, s_0, T) can be simulated by a cellular automaton $(1, S', \{-1, 0, 1\}, f)$ as follows. Let $S' = \Sigma \cup S \cup \{\bullet, \leftrightarrow\}$. A configuration (s, z, c) of the Turing machine is encoded as a configuration $c' = \tau(s, z, c)$ of the cellular automaton in the following way: for all $z' < z$, $c'(2z') = \bullet$, $c'(2z' + 1) = c(z)$; for all $z' > z$, $c'(2z' + 1) = \bullet$, $c'(2z' + 2) = c(z)$; $c'(2z) = \bullet$ and either $c'(2z + 1) = s$, $c'(2z + 2) = c(z)$

or $c'(2z+1) = c(z)$, $c'(2z+2) = s$ (two possible encodings). The local rule encode the transition function of the Turing machine. Applying the rule: for each transition $T(s,a) = (s', b, \leftarrow)$, $f(\bullet, s, a) = s'$, $f(s, a, \bullet) = b$, $f(\bullet, a, s) = s'$, $f(a, s, \bullet) = b$, for each transition $T(s,a) = (s', b, \rightarrow)$, $f(\bullet, s, a) = b$, $f(s, a, \bullet) = s'$, $f(\bullet, a, s) = b$, $f(a, s, \bullet) = s'$. Moving: for all $s \in S$, $a, b \in \Sigma$, $f(\leftrightarrow, s, a) = \bullet$, $f(a, \leftrightarrow, s) = s$ $f(a, s, \leftrightarrow) = \bullet$, $f(s, \leftrightarrow, a) = s$. All undefined transitions apply the identity rule but for $\bullet$ and $\leftrightarrow$ that alternates: for all states $x, y \in S$, $f(x, \bullet, y) = \leftrightarrow$ and $f(x, \leftrightarrow, y) = \bullet$ With this encoding, starting from an encoded configuration $\tau(\mathfrak{c})$, the configuration evolves in two steps to a configuration $\tau(\mathfrak{c}')$ where $\mathfrak{c}'$ is the next computation step of the Turing machine if it exists, $\mathfrak{c}' = \mathfrak{c}$ otherwise. Using this simulation, a Turing machine with m states and n symbols is simulated by a one-dimensional cellular automaton with first-neighbors and $m + n + 2$ states.

Simulation of tag systems is more tricky due to the non-locality of one computation step of the system. Following Cook [15], one can consider cyclic tag systems. A cyclic tag system is given as a finite set of words $(w_0, \ldots, w_{N-1})$ on the alphabet $\{\circ, \bullet\}$. A configuration of the system is a word $u \in \{\circ, \bullet\}^*$. At time step t, the configuration u evolves to a configuraton v according if either $u_0 = \circ$ and $u_0 v = u$, either $u_0 = \bullet$ and $u_0 v = u w_{t \bmod N}$. Cyclic tag systems can encode any recursive function. To encode all cyclic tag systems in a same cellular automaton $(1, S, \{-1, 0, 1\}, f)$, one can follow the following principle. Encode each configuration u of a cyclic tag system $(w_0, \ldots, w_{N-1})$ as a configuration of the kind ${}^{\omega}(T\_^k) \cdot u \cdot \blacksquare (w_0 \blacktriangle \ldots \blacktriangle w_{N-1})^{\omega}$ where intuitively T is a clock signal, $\blacksquare$ is the frontier between u and the rule and the rule is repeated on the right each word separated by a $\blacktriangle$. Giving the complete local rule is tedious but let us sketch its principle: each time a clock signal hits the first letter of u, it erases it and send a signal to the right transporting the value of that letter; when the signal meets the $\blacksquare$ it removes it and begins to treat the w word on the right, either erasing it or just crossing it; when the signal meets a $\blacktriangle$, it changes it into a $\blacksquare$ and the signal disappears. This principle is used by Cook to simulate cyclic tag systems with rule 110 particles and collisions.

A main point of discussion there is to decide which kind of configurations are acceptable for encoding Turing-universal computation. Finite configurations are certainly not a problem and using any, potentially non-recursive, configuration would permit trivial cellular automata to be misleadingly called universal. The previous constructions involving Turing machines use finite or ultimately periodic configurations with a same period on both sides, the same one for all simulated machines, whereas the tag system encoding uses ultimately periodic configurations with different periods, moreover these periodic parts depend on the simulated tag system. The tag system really needs this ultimate information, transforming the simulating cellular automaton into one working on finite configurations would have a constant but large impact on the number of states. As pointed in Durand and Róka [22], this configuration encoding problem adds difficulties to the formal definition of Turing-universality.

6. Intrinsic Universality

Even if the concept of intrinsically universal cellular automata took some time to emerge, intrinsic universality does not require more complex constructions to be achieved. Several technics are used to construct them:

Parallel Turing machines table lookup: A simple way to achieve intrinsic universality is to use synchronized parallel Turing heads (one copy of the same Turing machine per encoded cell) to lookup in the transition table (one copy in each encoded cell) of the encoded cellular automaton. Notice that the Turing machines used for this are not the same ones that are Turing-universal. In fact, their computational power is very small but they can carry precise information movement tasks.

One-way totalistic lookup: Another more cellular automata centric way to achieve intrinsic universality is, following Albert and Čulik 1987 [1], to simplify the task of the previous machine by simulating only one-way totalistic cellular automata which are sufficient to simulate all cellular automata.

Signals: The previous models are complex because the information displacement involved is still complex due to the sequential behavior of head-based machines. Following Ollinger [64] and Richard [71], particles and collisions, that is signals in the style of Mazoyer and Terrier [48], can be used to encode the information and perform the lookup task with parallel information displacement.

We explain here the parallel Turing machines table lookup technic, the other ones being refinements based on it. The one-dimensional first-neighbors universal cellular automaton $\mathcal{U}$ simulates a cellular automaton $(1, S, \{-1, 0, 1\}, f)$ the following way. Each configuration c is encoded as the concatenation of $\psi_{\mathcal{A}}(c(z))$ for all z. For each state $s \in S$, $\psi_{\mathcal{A}}(s)$ is a word of the kind $\blacksquare\tau(f(1,1,1)) \bullet \tau(f(1,1,2)) \bullet \ldots \bullet \tau(f(N,N,N))\blacktriangle 0^k\tau(s)0^k0^k$ where N is the size of S, k is the number of bits needed to encode numbers from 1 to N and $\tau(s)$ is a binary encoding of the state s. The simulation proceeds as follows so that the movement of each head is the same, up to translation, on each well encoded configuration. First the $\blacksquare$ letter is activated as a Turing head in initial state. The Turing head then moves to the left and copies the state $\tau(s_L)$ of the left neighbor in place of the first 0^k block. Then it symmetrically copies the state $\tau(s_R)$ of the right neighbor in place of the second 0^k block. This being done, the head scans the entire transition table, incrementing a counter (for example stored on top of the encoded states) at each step: if at some point the counter is equal to the triple of states, the result is copied from the transition table to the third 0^k block. At the end of the scan, the counter information is cleared, the result of the transition is copied in the $\tau(s)$ place and all three 0^k blocks are restored. The head then goes back to the $\blacksquare$. Using this simulation, one step of the simulated cellular automaton is simulated in a constant number of step for each cell by each Turing head. The universal automaton does not depend on the simulated automaton and is so intrinsically universal. A careful design can lead to less than 20 states. If the simulation uses the one-way totalistic technic, encoding states in unary, then it is easy to go under 10 states.

Notice that general Turing-universal cellular automata construction schemes from previous section concerning Turing machines can be adapted to produce small intrinsically universal cellular automata: apply the encoding schemes on machines performing the cell simulation task. However, the tag system simulations do not provide direct way to obtain intrinsic universality. Moreover, it is possible to design cellular automata Turing-universal by tag system simulation and not intrinsically universal.

More exotic intrinsically universal cellular automata have been studied on constrained rules. As an example, Moreira [54] constructed number conserving intrinsically universal automata, Bartlett and Garzon [6, 7] and Ollinger [62] do the same for bilinear cellular automata, and Theyssier [79] for captive cellular automata for which he proves that almost all captive automata are intrinsically universal.

7. Reversibility and Universality

Reversible cellular automata are special in the sense that they can achieve Turing-universality as any Turing machine can be simulated by a reversible Turing machine but they cannot achieve intrinsic universality: reversible cellular automata only simulate reversible cellular automata. However, there exists reversible cellular automata which are universal with respect to the class of reversible cellular automata.

Definition 7.1 (reversible intrinsic universality)**.** A reversible cellular automaton $\mathcal{U}$ is intrinsically universal for reversible cellular automata if for each reversible cellular automaton $\mathcal{A}$ of the same dimension there exists an unpacking map o_m, a positive integer $n \in \mathbb{N}$ and a translation vector $v \in \mathbb{Z}^d$ such that $G_{\mathcal{A}} \prec o_m^{-1} \circ G_{\mathcal{U}}^n \circ o_m \circ \sigma_v$.

Turing-universality and weak form of intrinsic universality have been proposed by Morita [56], Morita and Imai [58, 59], Durand-Lose [23, 24], Miller and Fredkin [50]. As for classical cellular automata in higher dimension the simulation of reversible boolean circuits automatically gives reversible intrinsic universality.

For one-dimensional cellular automata, reversible intrinsic universality can be achieved by simulating any one-way reversible partitioned reversible cellular automaton with a first-neigbors reversible partitioned reversible cellular automaton. We briefly sketch how to use a scheme similar to parallel Turing machine table lookup with reversible Turing machines to achieve this goal. The first adaptation is to remark that a the local partition rule of a reversible automaton is a permutation, thus it can be encoded as a finite sequence of permutation pairs. So, the table of transition is encoded as a finite sequence of pairs of states. The reversible Turing machine task is to scan the transition table and for each pair it contains to replace the actual state by the second element of the pair if the state appears in the current pair. It is technical but straightforward to see that a reversible Turing machine can achieve this. The information movement is reversible as in partitioned automata each cell gives half its state to a neighbor and take half a state from the other without erasing any information. Developing this simulation scheme, one constructs a reversible intrinsically universal cellular automaton.

8. Conclusion

This brief reading guide has given the reader the keys both to further explore the literature and to construct by itself conceptually simple Turing-universal and/or intrinsically universal cellular automata in one and two dimensions. The following broad bibliography can be explored with the help of the main text, all references being cited.

References

1. J. Albert and K. Čulik, II, *A simple universal cellular automaton and its one-way and totalistic version*, Complex Systems **1** (1987), no. 1, 1–16.
2. S. Amoroso and Y. N. Patt, *Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures*, Journal of Computer and System Sciences **6** (1972), 448–464.
3. M. A. Arbib, *Simple self-reproducing universal automata*, Information and Control **9** (1966), no. 2, 177–189.
4. E. R. Banks, *Universality in cellular automata*, Symposium on Switching and Automata Theory (Santa Monica, California, 1970), IEEE, 1970, pp. 194–215.
5. ______, *Information processing and transmission in cellular automata*, Ph.D. thesis, Massachusetts Institute of Technology, 1971.
6. R. Bartlett and M. Garzon, *Monomial cellular automata*, Complex Systems **7** (1993), no. 5, 367–388.
7. ______, *Bilinear cellular automata*, Complex Systems **9** (1995), no. 6, 455–476.
8. C. H. Bennett, *Logical reversibility of computation*, IBM Journal of Research and Development **17** (1973), no. 6, 525–532.
9. E. R. Berlekamp, J. H. Conway, and R. K. Guy, *Winning ways for your mathematical plays. Vol. 2*, Academic Press Inc. [Harcourt Brace Jovanovich Publishers], London, 1982, Games in particular.
10. A. W. Burks, *Programming and the theory of automata*, Essays on Cellular Automata (A. W. Burks, ed.), University of Illinois Press, Urbana, 1970, pp. 65–83 (Essay Two).
11. ______, *Towards a theory of automata based on more realistic primitive elements*, Essays on Cellular Automata (A. W. Burks, ed.), University of Illinois Press, Urbana, 1970, pp. 84–102 (Essay Three).
12. ______, *Von neumann's self-reproducing automata*, Essays on Cellular Automata (A. W. Burks, ed.), University of Illinois Press, Urbana, 1970, pp. 3–64 (Essay One).
13. J. Cocke and M. Minsky, *Universality of tag systems with p=2*, J. ACM **11** (1964), no. 1, 15–20.
14. E. F. Codd, *Cellular automata*, Academic Press, New York, 1968.
15. M. Cook, *Universality in elementary cellular automata*, Complex Systems **15** (2004), 1–40.
16. M. D. Davis, *A note on universal turing machines*, Automata Studies (C. E. Shannon and J. McCarthy, eds.), Princeton University Press, Princeton, 1956, pp. 167–175.
17. ______, *The universal computer: The road from leibniz to turing*, W. W. Norton & Company, 2000.
18. M. Delorme, *An introduction to cellular automata: some basic definitions and concepts*, Cellular automata (Saissac, 1996) (M. Delorme and J. Mazoyer, eds.), Kluwer Acad. Publ., Dordrecht, 1999, pp. 5–49.
19. J.C. Delvenne, P. Kurka, and V. D. Blondel, *Decidability and universality in symbolic dynamical systems*, Fundam. Inform. **74** (2006), no. 4, 463–490.
20. A. K. Dewdney, *The cellular automata programs that create wireworld, rugworld and other diversions*, Scientific American **262** (1990), 146–149.
21. Jean-Christophe Dubacq, *How to simulate turing machines by invertible one-dimensional cellular automata*, Int. J. Found. Comput. Sci. **6** (1995), no. 4, 395–402.
22. B. Durand and Zs. Róka, *The game of life: universality revisited*, Cellular automata (Saissac, 1996) (M. Delorme and J. Mazoyer, eds.), Kluwer Acad. Publ., Dordrecht, 1999, pp. 51–74.
23. J. Durand-Lose, *Reversible cellular automaton able to simulate any other reversible one using partitioning automata*, LATIN '95 (R. A. Baeza-Yates, E. Goles Ch., and P. V. Poblete, eds.), Lecture Notes in Computer Science, vol. 911, Springer, 1995, pp. 230–244.
24. ______, *Automates cellulaires, automates partitions et tas de sable*, Ph.D. thesis, Université Bordeaux I, 1996.
25. ______, *Intrinsic universality of a 1-dimensional reversible cellular automaton*, STACS 97 (Lübeck), Lecture Notes in Comput. Sci., vol. 1200, Springer, Berlin, 1997, pp. 439–450.
26. Patrick C. Fischer, *On formalisms for turing machines*, J. ACM **12** (1965), no. 4, 570–580.
27. E. Fredkin and T. Toffoli, *Conservative logic*, International Journal of Theoretical Physics **21** (1982), 219–253.
28. A. Gajardo and E. Goles Ch., *Universal cellular automaton over a hexagonal tiling with 3 states*, IJAC **11** (2001), no. 3, 335–354.
29. M. Gardner, *Mathematical games: The fantastic combinations of John Conway's new solitaire game 'Life'*, Scientific American **223** (1970), no. 4, 120–123.

30. K. Gödel, *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I*, Monatschefte für Mathematik und Physik **38** (1931), 173–198, reprinted in S. Feferman, W. Dawson, S.C. Kleene, G. Moore, R.M. Solovay, J. van Heijendort (eds.), *Kurt Gödel: Collected Works*, Oxford University Press, Oxford, **1** (1986), 144–195.
31. G. A. Hedlund, *Endormorphisms and automorphisms of the shift dynamical system*, Mathematical Systems Theory **3** (1969), 320–375.
32. Hertling, *Embedding cellular automata into reversible ones*, Unconventional Models of Computation, Springer, 1998 (C. S. Calude, J. Casti, and M. J. Dinneen, eds.), 1998.
33. K. Imai and K. Morita, *A computation-universal two-dimensional 8-state triangular reversible cellular automaton*, Theoretical Computer Science **231** (2000), no. 2, 181–191.
34. J. Kari, *Reversibility of 2D cellular automata is undecidable*, Physica D. Nonlinear Phenomena **45** (1990), no. 1-3, 379–385, Cellular automata: theory and experiment (Los Alamos, NM, 1989).
35. ______, *Reversibility and surjectivity problems of cellular automata*, J. Comput. Syst. Sci. **48** (1994), no. 1, 149–182.
36. ______, *Theory of cellular automata: A survey*, Theoretical Computer Science **334** (2005), 3–33.
37. S. C. Kleene, *Representation of events in nerve nets and finite automata*, Automata Studies (C. E. Shannon and J. McCarthy, eds.), Princeton University Press, Princeton, 1956, pp. 3–41.
38. G. Lafitte, *Gödel incompleteness revisited*, personal communication *(submitted to JAC 2008)*, 2008.
39. C.G. Langton, *Self-reproduction in cellular automata*, Physica D **10** (1984), no. 1-2, 135–144.
40. Y. Lecerf, *Machines de turing réversibles*, C. R. Acad. Sci. Paris **257** (1963), 2597–2600.
41. K. Lindgren and M. G. Nordahl, *Universal computation in simple one-dimensional cellular automata*, Complex Systems **4** (1990), no. 3, 299–318.
42. N. Margolus, *Physics-like models of computation*, Physica D **10** (1984), 81–95.
43. B. Martin, *Construction modulaire d'automates cellulaires*, Ph.D. thesis, École Normale Supérieure de Lyon, 1993.
44. B. Martin, *A universal cellular automaton in quasilinear time and its S-m-n form*, Theoretical Computer Science **123** (1994), no. 2, 199–237.
45. J. Mazoyer, *Computations on one dimensional cellular automata*, Ann. Math. Artif. Intell. **16** (1996), 285–309.
46. ______, *Computations on grids*, Cellular automata (Saissac, 1996), Kluwer Acad. Publ., Dordrecht, 1999, pp. 119–149.
47. J. Mazoyer and Ivan Rapaport, *Inducing an order on cellular automata by a grouping operation*, Discrete Applied Mathematics **91** (1999), no. 1-3, 177–196.
48. J. Mazoyer and Véronique Terrier, *Signals in one-dimensional cellular automata*, Theoretical Computer Science **217** (1999), no. 1, 53–80, Cellular automata (Milan, 1996).
49. W. S. McCulloch and W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics **5** (1943), 115–133.
50. D. B. Miller and E. Fredkin, *Two-state, reversible, universal cellular automata in three dimensions*, Conf. Computing Frontiers (N. Bagherzadeh, M. Valero, and A. Ramírez, eds.), ACM, 2005, pp. 45–51.
51. M. Minsky, *Computation: Finite and infinite machines*, Prentice Hall, Englewoods Cliffs, 1967.
52. E. F. Moore, *Machine models of self-reproduction*, Proceedings of Symposia in Applied Mathematics, vol. 14, American Mathematical Society, 1962, pp. 17–33.
53. ______, *Machine models of self-reproduction*, Essays on Cellular Automata (A. W. Burks, ed.), University of Illinois Press, Urbana, 1970, pp. 187–203 (Essay Six).
54. A. Moreira, *Universality and decidability of number-conserving cellular automata*, Theoretical Computer Science **292** (2003), no. 3, 711–721.
55. K. Morita, *A simple construction method of a reversible finite automaton out of fredkin gates, and its related problem*, IEICE Trans. Inf. & Syst. **E73** (1990), no. 6, 978–984.
56. ______, *Reversible simulation of one-dimensional irreversible cellular automata*, Theoretical Computer Science **148** (1995), no. 1, 157–163.
57. K. Morita and M. Harao, *Computation universality of one-dimensional reversible (injective) cellular automata*, IEICE Trans. Inf. & Syst. **E72** (1989), 758–762.
58. K. Morita and K. Imai, *Self-reproduction in a reversible cellular space*, Theoretical Computer Science **168** (1996), no. 2, 337–366.

59. ______, *Number-conserving reversible cellular automata and their computation-universality*, Theoretical Informatics and Applications **35** (2001), no. 3, 239–258.
60. T. Neary and D. Woods, *P-completeness of cellular automaton rule 110*, Proceedings of ICALP 2006, Lecture Notes in Computer Science, vol. 4051, Springer, Berlin, 2006, pp. 132–143.
61. F. Noural and R.S. Kashef, *A universal four-state cellular computer*, IEEE Transactions on Computers **24** (1975), no. 8, 766–776.
62. N. Ollinger, *Two-states bilinear intrinsically universal cellular automata*, Fundamentals of computation theory (Riga, Latvia, 2001) (Berlin) (R. Freivalds, ed.), Lecture Notes in Computer Science, vol. 2138, Springer, 2001, pp. 396–399.
63. ______, *Automates cellulaires : structures*, Ph.D. thesis, École Normale Supérieure de Lyon, 2002.
64. ______, *The quest for small universal cellular automata*, International Colloquium on Automata, languages and programming (Málaga, Spain, 2002) (Berlin) (P. Widmayer, F. Triguero, R. Morales, M. Hennessy, S. Eidenbenz, and R. Conejo, eds.), Lecture Notes in Computer Science, vol. 2380, Springer, 2002, pp. 318–329.
65. ______, *The intrinsic universality problem of one-dimensional cellular automata*, Symposium on Theoretical Aspects of Computer Science (Berlin, Germany, 2003), Lecture Notes in Computer Science, Springer, Berlin, 2003, (to appear).
66. D. Perrin, *Les débuts de la théorie des automates*, Technique et Science Informatique **14** (1995), 409–433.
67. E. Post, *The two-valued iterative systems of mathematical logic*, Princeton University Press, Princeton, 1941.
68. ______, *Formal reductions of the general combinatorial decision problem*, American Journal of Mathematics **65** (1943), no. 2, 197–215.
69. I. Rapaport, *Inducing an order on cellular automata by a grouping operation*, Ph.D. thesis, École Normale Supérieure de Lyon, 1998.
70. G. Richard, *From turing machine to rule 110: embedding computational power in small cellular automata through particles and collisions*, personal communication *(submitted to JAC 2008)*, 2008.
71. ______, *A particular universal cellular automaton*, personal communication, 2008.
72. D. Richardson, *Tessellations with local transformations*, Journal of Computer and System Sciences **6** (1972), 373–388.
73. C. E. Shannon, *A universal turing machine with two internal states*, Automata Studies (C. E. Shannon and J. McCarthy, eds.), Princeton University Press, Princeton, 1956, pp. 157–165.
74. A. R. Smith, III, *Simple computation-universal cellular spaces*, Journal of the ACM **18** (1971), 339–353.
75. K. Sutner, *Universality and cellular automata*, MCU 2004 (M. Margenstern, ed.), Lecture Notes in Computer Science, vol. 3354, Springer, 2004, pp. 50–59.
76. J. W. Thatcher, *Self-describing turing machines and self-reproducing cellular automata*, Essays on Cellular Automata (A. W. Burks, ed.), University of Illinois Press, Urbana, 1970, pp. 103–131 (Essay Four).
77. ______, *Universality in the von neumann cellular model*, Essays on Cellular Automata (A. W. Burks, ed.), University of Illinois Press, Urbana, 1970, pp. 132–186 (Essay Five).
78. G. Theyssier, *Automates cellulaires : un modèle de complexités*, Ph.D. thesis, École Normale Supérieure de Lyon, 2005.
79. ______, *How common can be universality for cellular automata?*, STACS 2005, Lecture Notes in Computer Science, vol. 3404, Springer, 2005, pp. 121–132.
80. Tommaso Toffoli, *Computation and construction universality of reversible cellular automata*, J. Comput. Syst. Sci. **15** (1977), no. 2, 213–231.
81. A. M. Turing, *On computable numbers with an application to the entscheidungs problem*, Proceedings of the London Mathematical Society 2 **42** (1936), 230–265.
82. J. von Neumann, *Theory of self-reproducing automata*, University of Illinois Press, Urbana, Ill., 1966, (A. W. Burks, ed.).
83. S. Wolfram, *Universality and complexity in cellular automata*, Physica D. Nonlinear Phenomena **10** (1984), no. 1-2, 1–35, Cellular automata (Los Alamos, N.M., 1983).
84. S. Wolfram, *A new kind of science*, Wolfram Media Inc., Champaign, Ilinois, US, United States, 2002.
85. D. Woods and T. Neary, *On the time complexity of 2-tag systems and small universal turing machines*, FOCS 2006, IEEE Computer Society, 2006, pp. 439–448.
86. ______, *The complexity of small universal turing machines*, Computability in Europe (S. B. Cooper, B. Löwe, and A. Sorbi, eds.), Lecture Notes in Computer Science, vol. 4497, Springer, 2007, pp. 791–799.

Journées Automates Cellulaires 2008 (Uzès), pp. 119-129

THE SPATIAL STRUCTURE OF ODOMETERS IN CERTAIN CELLULAR AUTOMATA

MARCUS PIVATO [1] AND REEM YASSAWI [1]

E-mail address: marcuspivato@trentu.ca,ryasssawi@trentu.ca

[1] Department of Mathematics, Trent University, Peterborough, Ontario, Canada

ABSTRACT. Recent work has shown that many cellular automata (CA) have configurations whose orbit closures are isomorphic to odometers. We investigate the geometry of the spacetime diagrams of these 'odometer configurations'. For boolean linear CA, we exactly determine the positions of the consecutive 'gears' of the odometer mechanism in the configuration. Then we characterize and explain the self-similar structure visible in the spacetime diagrams of odometer configurations for two classes of nonlinear CA: ratchet CA and Coven CA.

1. Introduction

Let $\mathcal{A}$ be a finite alphabet. If $\Phi : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}}$ is a cellular automaton (CA), with left radius 0, then Φ can also be treated as a one-sided CA $\Phi_{\mathbb{N}} : \mathcal{A}^{\mathbb{N}} \longrightarrow \mathcal{A}^{\mathbb{N}}$. In [CPY07], the authors showed:

Theorem 1.1. *Let $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ be a left permutative CA with left radius 0. If $\mathbf{z} \in \mathcal{A}^{\mathbb{N}}$ is $\Phi_{\mathbb{N}}$-periodic, $\mathbf{x} = \mathbf{y}.\mathbf{z} \in \mathcal{A}^{\mathbb{Z}}$ and the Φ-orbit $\mathcal{O}_\Phi(\mathbf{x}) := \{\Phi^t(\mathbf{x})\}_{t=0}^{\infty}$ is infinite, then the orbit closure $(\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi)$ is topologically conjugate to an odometer.*

(See §2 for definitions and notation). In this article, we discuss which odometers can be embedded in certain linear cellular automata, and the physical bounds on how these odometers are embedded. We also investigate the self similarity of the spacetime diagrams which display these odometers, in some linear and also non linear cellular automata. We start by generalising a result in [CY07] concerning which odometers can be embedded in the Ledrappier CA:

Proposition 1.2. *Suppose that the set of infinite multiplicity prime divisors of the quotient set $\mathcal{Q}$ are $\{q_1, q_2, \ldots q_n\}$, and let $\tau : \mathcal{Z}(\mathcal{Q}) \to \mathcal{Z}(\mathcal{Q})$ be an odometer. Let $N = \Pi_{j=1}^{n} q_j$ and let $\mathcal{A} = \mathbb{Z}_{/N}$. Then τ embeds in any linear CA $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ with $\Phi(\mathbf{x}) = \mathbf{x} + \sum_{i=1}^{r} a_i \sigma^i(\mathbf{x})$, where for each $j \in \{1, \ldots, n\}$, at least one of the a_i's does not divide q_j. Furthermore no other odometer can be embedded in Φ.*

2000 ACM Subject Classification: F.1.1.

Key words and phrases: linear cellular automata; dynamics; Coven cellular automata; odometer; self-similar; substitution.

With the conditions on $\mathbf{x}$ in Theorem 1.1, left permutativity of Φ implied that the columns of the Φ-spacetime diagram of $\mathbf{x}$, $\mathbf{ST}_\Phi(\mathbf{x})$, were all periodic, and the fact that $\mathbf{x}$ had an infinite Φ-orbit meant that there was a sequence of columns $\mathbf{C}_{k_n}$ in $\mathbf{ST}_\Phi(\mathbf{x})$ whose periodicities $\{p_n\}$ increased to infinity. A conjugacy was then constructed between $\overline{\mathcal{O}_\Phi}(\mathbf{x})$ and the odometer with quotient set $\mathcal{Q} := \{\ldots, \frac{p_3}{p_2}, \frac{p_2}{p_1}, \frac{p_1}{p}, p\}$. For many Φ, and many initial points $\mathbf{x}$, $\mathbf{ST}_\Phi(\mathbf{x})$ had a clear self-similar structure, reminiscent of the Sierpinski gasket - this is of course known if Φ is linear, but it also turned out to be the case for many nonlinear CA, such as some Coven CA. This fact led to interest in analysing these spacetime diagrams, and the rigidity imposed on them by the odometers that $(\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi)$ are. One result in this direction is

Theorem 1.3. *Let $\Phi(\mathbf{x}) = \mathbf{x} + \sum_{p=1}^{L} \sigma^{a_p}(\mathbf{x})$ be defined on ${\mathbb{Z}_{/2}}^{\mathbb{Z}}$, with $0 < a_1 < a_2 < \ldots a_L$. Suppose that $\mathbf{x}_{[0\ldots\infty)}$ is $\Phi_{\mathbb{N}}$-fixed, but $\mathbf{x}_{[-1\ldots\infty)}$ is not $\Phi_{\mathbb{N}}$-fixed. Then, letting $\mathbf{ST}_\Phi(\mathbf{x}) \in \mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$ be the Φ-spacetime diagram of $\mathbf{x}$, the sequence $\{\mathbf{C}_{k_n}\}_{n\geq 1}$ of columns, where the periodicity first jumps to 2^n, are those where $\{k_n\} = \{2^n a_1 - a_1 + 1\}_{n\geq 1}$.*

Theorem 1.3 implies that $\mathcal{O}_\Phi(\mathbf{x})$ is infinite; then Theorem 1.1 says that $(\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi)$ is conjugate to any odometer that Theorem 1.2 allows. The columns $\{\mathbf{C}_{k_n}\}_{n\geq 1}$ can be thought of as the 'gears' of this odometer mechanism, and Theorem 1.3 says that the location of these gears is determined entirely by Φ, and is independent of $\mathbf{x}$. The odometer structures of linear CA often exhibit self-similar spacetime diagrams (see §3); Theorem 1.3 forces the 'scaling factor' of this self-similarity to be independent of the initial point generating the self similar diagram. Some version of this theorem may well be true for some linear Φ defined on larger alphabets. However the condition that $\Phi(\mathbf{x}) = \mathbf{x}+$ [something] cannot be relaxed. For example, if $\Phi(\mathbf{x}) = 2x + \sigma\mathbf{x}$ on $(\mathbb{Z}_{/3})^{\mathbb{Z}}$, and if $\mathbf{x}_{[-1\ldots\infty)} = [1, 0, 0, 0, \ldots]$, then $\Phi(\mathbf{x}_{[0\ldots\infty)}) = \mathbf{x}_{[0\ldots\infty)}$, the column $\mathbf{C}_{-1}$ in $\mathbf{ST}_\Phi(\mathbf{x})$ has period 2, and $\mathbf{C}_{-2}$ has period 6, irrespective of the choice of x_{-2}. However, the choice of x_{-2} affects the periodicity of $\mathbf{C}_{-3}$: if $x_{-2} = 2$, then $\mathbf{C}_{-3}$ has period 6; otherwise it has period 18.

In §3, self-similarity of $\mathbf{ST}_\Phi(\mathbf{x})$ is defined in terms of two-dimensional substitution systems. We consider two classes of CA: the $\mathbb{Z}_{/n}$-ratchet CA, and the range-R Coven CA (both nonlinear generalisations of the Ledrappier CA). We prove that for these CA, there exist points $\mathbf{x}$ such that $\mathbf{ST}_\Phi(\mathbf{x})$ is self-similar (Propositions 3.7 and 3.9). In addition to being intrinsically interesting, the self-similarity allows us to easily characterize the spatiotemporal structure of these odometers. Finally, an Appendix contains most of the proofs.

2. Preliminaries

Cellular automata. Let $\mathcal{A}$ be a finite alphabet, and let $\mathcal{A}^{\mathbb{Z}}$ be the space of all doubly infinite sequences with entries from $\mathcal{A}$. Elements $\mathbf{x} \in \mathcal{A}^{\mathbb{Z}}$ will sometimes be written as $\mathbf{x} = \mathbf{y}.\mathbf{z}$, where $\mathbf{y}$, $\mathbf{z} \in \mathcal{A}^{\mathbb{N}^+}$, $\mathcal{A}^{\mathbb{N}}$ respectively. A *cellular automaton* (CA) is a continuous, shift-commuting self-map $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$. It is known that every CA Φ is given by a local rule $\phi : \mathcal{A}^{[-l\ldots r]} \longrightarrow \mathcal{A}$, for some l, $r \geq 0$ (the *left and right radii* of Φ), such that for all $\mathbf{x} \in \mathcal{A}^{\mathbb{Z}}$, and all $i \in \mathbb{Z}$,

$$[\Phi(\mathbf{x})]_i = \phi(x_{i-l}, x_{i-l+1}, \ldots, x_{i+r}) .$$

If l or r are 0, then Φ can also act on sequences from $\mathcal{A}^{\mathbb{N}}$. Φ is *left permutative* if for every $x_1, \ldots x_r \in \mathcal{A}$, the map $\phi(\bullet, x_1, \ldots, x_r)$ is a permutation of $\mathcal{A}$, similarly for *right permutative.* Let $\mathbb{Z}_{/p}$ be the cyclic group of p elements. The CA Φ is *linear* if Φ is a group endomorphism of $(\mathbb{Z}_{/p})^{\mathbb{Z}}$, where the group operation is componentwise addition. Φ can then be written as $\Phi(\mathbf{x}) = \sum_{i=0}^{r} a_i\sigma^i(\mathbf{x})$ where σ is the left shift map on $(\mathbb{Z}_{/p})^{\mathbb{Z}}$. If $x \in \mathcal{A}^{\mathbb{Z}}$, the Φ-*spacetime diagram* of $\mathbf{x}$, $\mathbf{ST}_{\Phi}(\mathbf{x})$, is the element in $\mathcal{A}^{\mathbb{N}\times\mathbb{Z}}$ whose kth row is $\Phi^k(\mathbf{x})$. For any integer n, we let $\mathbf{C}_n := \{(\Phi^k(\mathbf{x}))_n\}_{k\geq 0}$.

Odometers. Let $\mathcal{Q} := (q_1, q_2, \ldots q_n)$ be an ordered set (or sequence, if $n = \infty$) of integers ≥ 2 (the *quotient set*). Let $\mathcal{Z}(\mathcal{Q}) := \prod_{l=1}^{n} \mathbb{Z}_{q_l}$ be the Cartesian product set. $(\mathcal{Z}(\mathcal{Q}), \oplus)$ is a (compact, abelian) group where "$\oplus$" is defined as addition "with carry": if $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots y_n)$, then $\mathbf{x} \oplus \mathbf{y} := (r_1, r_2, \ldots r_n)$ where $x_1 + y_1 = k_1 q_1 + r_1$, and inductively, $\sum_{l=1}^{n-1} k_l + x_n + y_n = k_n q_n + r_n, k_l + x_n + y_n = k_n q_n + r_n$, for $n > 1$, with $k_n \geq 0$ and $0 \leq r_n < q_n$ for each n.

$\mathbf{g} \in \mathcal{Z}(\mathcal{Q})$ is a *topological generator* for $\mathcal{Z}(\mathcal{Q})$ if $\{n\,\mathbf{g}\}_{n\in\mathbb{N}}$ is dense in $\mathcal{Z}(\mathcal{Q})$. For such $\mathbf{g}$ we define the $\mathbf{g}$-*odometer* $\tau_{\mathbf{g}} : \mathcal{Z}(\mathcal{Q}) \to \mathcal{Z}(\mathcal{Q})$ as $\tau_{\mathbf{g}}(\mathbf{z}) = \mathbf{z} \oplus \mathbf{g}, \quad \forall\ \mathbf{z} \in \mathcal{Z}(\mathcal{Q})$. If $\mathbf{g} = (g_1, g_2, \ldots) \in \mathcal{Z}(\mathcal{Q})$ then $\{n\,\mathbf{g}\}_{n\in\mathbb{N}}$ is dense in $\mathcal{Z}(\mathcal{Q})$ if and only if for each n, the n-tuple $(g_1, g_2, \ldots, g_n)$ is a generator for the finite cyclic group $(\mathcal{Z}(\{q_1, q_2, \ldots, q_n\}), \oplus)$. If $\mathbf{g}$ and $\mathbf{g}^*$ are both topological generators for $\mathcal{Z}(\mathcal{Q})$, then $(\mathcal{Z}(\mathcal{Q}), \tau_{\mathbf{g}})$ is topologically conjugate to $(\mathcal{Z}(\mathcal{Q}), \tau_{\mathbf{g}^*})$. Thus we will assume that the generator is $\mathbf{1} = (1, 0, 0, \ldots)$ and write the odometer $\tau_{\mathbf{g}}$ as τ.

If p is prime, then the *multiplicity* of p in $\mathcal{Q}$ is the sum number of times (possibly infinite) that p occurs in the prime decomposition of the elements of sequence $\mathcal{Q}$.

Theorem 2.1. [BS95] *$(\mathcal{Z}(\mathcal{Q}), \tau)$ and $(\mathcal{Z}(\mathcal{Q}^*), \tau)$ are topologically conjugate if and only if for every prime p, the multiplicity of p in $\mathcal{Q}$ and $\mathcal{Q}^*$ is equal.*

Theorem 2.1 also tells us that we can assume that all elements in $\mathcal{Q}$ are prime. When convenient we will assume this. If $\mathcal{Q} = (p, p, \ldots)$, then let $\mathcal{Z}(p) := \mathcal{Z}(\mathcal{Q})$ (this is the group of *p-adic integers*). We consider quotient sets $\mathcal{Q}$ for whom only finitely many primes have positive multiplicity.

3. Self-similar structures in spacetime diagrams

Spacetime diagrams of linear CA often exhibit self-similar structures, as in Figure 1. This self-similarity reflects the self-similarity of Pascal's Triangle in $\mathbb{Z}_{/p}$, as described by Lucas' theorem [Luc78], and has been intensively studied [Wil87, Tak93, vHPS01, AvHP$^+$97, BvHPS03]. Self-similar structures also arise in *nonlinear* CA [e.g. see Figures 2 and 3 below], but these *cannot* be explained using Lucas' theorem. In this section we will develop an analytic framework to understand this self-similarity as 'compatibility' of the CA with a suitable substitution mappings on $\mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$.

Let $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ be a CA, with $\phi : \mathcal{A}^{\{0,1\}} \to \mathcal{A}$, and let $\mathbf{a} \in \mathcal{A}^{\mathbb{Z}}$. The *spacetime subshift* of Φ is the set $\mathcal{ST}(\Phi) \subseteq \mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$ of all spacetime diagrams of Φ. In other words, $\mathcal{ST}(\Phi)$ is the two-dimensional subshift of finite type in $\mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$, generated by the set of *admissible triominos*

$$\left\{ \begin{bmatrix} a\ b \\ c \end{bmatrix} \ ;\ a, b \in \mathcal{A};\ c = \phi(a, b) \right\}. \tag{3.1}$$

For example, Figures 1, 2 and 3 show spacetime diagrams, all exhibiting self-similarity. We will now explain this using the theory of substitution systems.

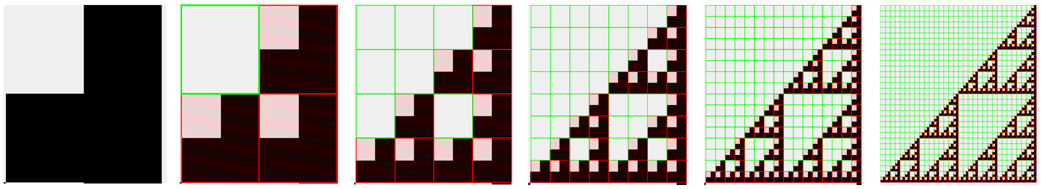


Figure 1: A self-similar spacetime diagram for the 'Ledrappier' CA on $\mathcal{A} = \mathbb{Z}_{/2}$ with local rule $\phi(x_0, x_1) = x_0 + x_1$. The five images show the same spacetime diagram on larger and larger scales. Each diagram can be obtained from the previous one by applying the substitution rule described in Examples 3.1. See also Examples 3.3 and 3.4.

Substitution configurations. Let $W, H \in \mathbb{N}$ and let $\mathcal{A}$ be a finite alphabet. A $W \times H$ *substitution rule* is a function $\varsigma : \mathcal{A} \longrightarrow \mathcal{A}^{W \times H}$. This defines a function $\boldsymbol{\varsigma} : \mathcal{A}^{\mathbb{Z}\times\mathbb{N}} \longrightarrow \mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$ where

$$\boldsymbol{\varsigma}\left(\begin{array}{cc|ccc} \cdots & b^0_{-1} & b^0_0 & b^0_1 & \cdots \\ \cdots & b^1_{-1} & b^1_0 & b^1_1 & \cdots \\ \cdots & b^2_{-1} & b^2_0 & b^2_1 & \cdots \\ \iddots & \vdots & \vdots & \vdots & \ddots \end{array}\right) := \begin{array}{cc|ccc} \cdots & \varsigma(b^0_{-1}) & \varsigma(b^0_0) & \varsigma(b^0_1) & \cdots \\ \cdots & \varsigma(b^1_{-1}) & \varsigma(b^1_0) & \varsigma(b^1_1) & \cdots \\ \cdots & \varsigma(b^2_{-1}) & \varsigma(b^2_0) & \varsigma(b^2_1) & \cdots \\ \iddots & \vdots & \vdots & \vdots & \ddots \end{array} \qquad (3.2)$$

(the lines indicate the positions of the axes). If $a \in \mathcal{A}$ and $n \in \mathbb{N}$, then we likewise define $\boldsymbol{\varsigma}^n(a) \in \mathcal{A}^{W^n \times H^n}$ in the obvious way. The *language* of ς is the set $\mathcal{L}(\varsigma)$ of all $n \times m$ blocks (for any $n, m \in \mathbb{N}$) which occur in $\boldsymbol{\varsigma}^k(a)$ for some $a \in \mathcal{A}$ and $k \in \mathbb{N}$. The ς*-substitution shift* $\mathfrak{Sub}(\varsigma)$ is the subshift of $\mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$ defined by $\mathcal{L}(\varsigma)$ [Que87, Fog02]. If Φ is a CA, we say that ς is *compatible* with Φ if $\mathfrak{Sub}(\varsigma) \subseteq \mathcal{ST}(\Phi)$.

EXAMPLE 3.1: Let $\mathcal{A} = \mathbb{Z}_{/2}$, and define $\varsigma : \mathcal{A} \longrightarrow \mathcal{A}^{2\times 2}$ by $\varsigma(0) = \left[\begin{smallmatrix} 0 & 0 \\ 0 & 0 \end{smallmatrix}\right]$; and $\varsigma(1) = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 1 \end{smallmatrix}\right]$. Then Figure 1 shows an element of $\mathfrak{Sub}(\varsigma)$. This is also the spacetime diagram generated by the 'Ledrappier' CA with local rule $\phi(x_0, x_1) = x_0 + x_1$. This suggests that ς is compatible with Φ. ◇

A ς*-seed* is a pair $[a, b] \in \mathcal{A}^2$ such that:

(i): $[a, b] \in \mathcal{L}(\varsigma)$; **(ii):** For all $c \in \mathcal{A}$, $\exists\ n \in \mathbb{N}$ such that c occurs in $\boldsymbol{\varsigma}^n[a, b]$.

(iii): $\varsigma(a) = \begin{bmatrix} * & \cdots & * & a \\ * & \cdots & * & * \\ \vdots & \iddots & \vdots & \vdots \\ * & \cdots & * & * \end{bmatrix}$; **(iv):** $\varsigma(b) = \begin{bmatrix} b & * & \cdots & * \\ * & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & * \end{bmatrix}$;

Note that, since $\mathcal{L}(\varsigma) = \mathcal{L}(\varsigma^n)$ for any natural n, then it is always possible to find a pair $[a, b]$ satisfying **(i)**, **(iii)** and **(iv)**, by the pigeonhole principle. Define $\mathbf{A} \in \mathcal{A}^{(-\infty\ldots 0]\times\mathbb{N}}$ by the property that $\mathbf{A}_{(-W^n\ldots 0]\times[0\ldots H^n)} = \boldsymbol{\varsigma}^n(a)$ for all $n \in \mathbb{N}$ (this definition is consistent because $\varsigma(a)^0_0 = a$). Likewise, define $\mathbf{B} \in \mathcal{A}^{[1..\infty)\times\mathbb{N}}$ by the property that $\mathbf{B}_{(0\ldots W^n]\times[0\ldots H^n)} = \boldsymbol{\varsigma}^n(b)$ for all $n \in \mathbb{N}$ (this definition is consistent because $\varsigma(b)^0_1 = b$). We write "$\mathbf{A} := \boldsymbol{\varsigma}^\infty(a)$" and "$\mathbf{B} := \boldsymbol{\varsigma}^\infty(b)$". Let $\boldsymbol{\varsigma}^\infty[a, b] := \overline{\mathbf{A}|\mathbf{B}}$ be the obvious element of $\mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$.

If $\mathbf{A} \in \mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$ and $n \in \mathbb{N}$, then the nth *row* of $\mathbf{A}$ is the biinfinite sequence $[\ldots\ a^n_{-1},\ a^n_0,\ a^n_1,\ \ldots]$. If $\mathbf{r} = [r_1, r_2] \in \mathcal{A}^2$, we say that $\mathbf{r}$ *occurs* in $\mathbf{A}$ if there is some $z \in \mathbb{Z}$ and $n \in \mathbb{N}$ such that $a^n_z = r_1$ and $a^n_{z+1} = r_2$. Let $\mathcal{L}_2(\varsigma) := \left\{\mathbf{r} \in \mathcal{A}^2 \ ;\ \mathbf{r} \text{ occurs in some } \mathbf{A} \in \mathfrak{Sub}(\varsigma)\right\}$. A configuration $\mathbf{S} \in \mathcal{A}^{\mathbb{Z}\times\mathbb{N}}$ is $\boldsymbol{\varsigma}$*-fixed* if $\boldsymbol{\varsigma}(\mathbf{S}) = \mathbf{S}$.

Lemma 3.2. *Let ς be a substitution and let $\mathbf{s} \in \mathcal{A}^2$ be a ς-seed. Then*

(a) $\mathbf{S} := \boldsymbol{\varsigma}^\infty(\mathbf{s})$ *is a $\boldsymbol{\varsigma}$-fixed configuration, and $\mathfrak{Sub}(\varsigma)$ is the σ-orbit closure of* $\mathbf{S}$.

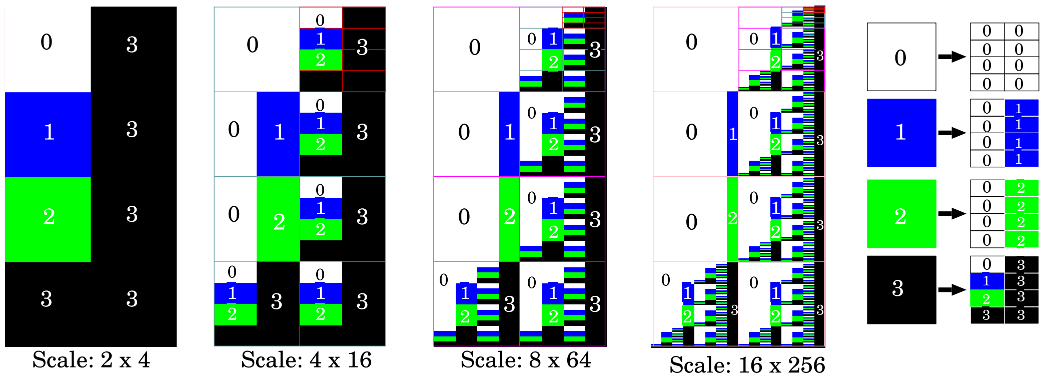


Figure 2: Self-similarity in the $\mathbb{Z}_{/4}$-ratchet CA. The left four images are the same spacetime diagram, shown on larger and larger scales. The numerical labels show how each spacetime diagram can be obtained from the previous one by applying the substitution mapping illustrated on the far right.

(b) *There exists $n \in \mathbb{N}$ so that $\mathcal{L}_2(\varsigma)$ is the set of all 2-words which occur in $\varsigma^n(\mathbf{s})$.*

Proof. See Appendix. □

EXAMPLE 3.3: Let $\mathcal{A} = \mathbb{Z}_{/2}$, and define $\varsigma : \mathcal{A} \longrightarrow \mathcal{A}^{2\times 2}$ as in Example 3.1 Then $[1,0]$ is a ς-seed. ♢

We say that ϕ *commutes with* ς if, for every $[a,b] \in \mathcal{L}_2(\varsigma)$ with $c = \phi(a,b)$,

$$\varsigma\begin{bmatrix} a\ b \\ c \end{bmatrix} := \begin{bmatrix} \varsigma(a)\ \varsigma(b) \\ \varsigma(c) \end{bmatrix} \quad \text{is a fragment of a spacetime diagram of } \Phi.$$

EXAMPLE 3.4: Let $\varsigma : \mathcal{A} \longrightarrow \mathcal{A}^{2\times 2}$ be as in Examples 3.1 and 3.3. The Ledrappier CA (with local rule $\phi(x_0, x_1) = x_0 + x_1$) commutes with ς, as shown by the following computations

$$\varsigma\begin{bmatrix} 0\ 0 \\ 0 \end{bmatrix} = \begin{array}{|c|c|} \hline 0\ 0 & 0\ 0 \\ 0\ 0 & 0\ 0 \\ \hline 0\ 0 & (0) \\ 0\ 0 & \\ \hline \end{array} \quad \varsigma\begin{bmatrix} 0\ 1 \\ 1 \end{bmatrix} = \begin{array}{|c|c|} \hline 0\ 0 & 0\ 1 \\ 0\ 0 & 1\ 1 \\ \hline 0\ 1 & (0) \\ 1\ 1 & \\ \hline \end{array} \quad \varsigma\begin{bmatrix} 1\ 0 \\ 1 \end{bmatrix} = \begin{array}{|c|c|} \hline 0\ 1 & 0\ 0 \\ 1\ 1 & 0\ 0 \\ \hline 0\ 1 & (0) \\ 1\ 1 & \\ \hline \end{array} \quad \varsigma\begin{bmatrix} 1\ 1 \\ 0 \end{bmatrix} = \begin{array}{|c|c|} \hline 0\ 1 & 0\ 1 \\ 1\ 1 & 1\ 1 \\ \hline 0\ 0 & (0) \\ 0\ 0 & \\ \hline \end{array}.$$

Observe that the image of each Φ-admissible triomino is a fragment of a Φ-spacetime diagram (to illustrate this, we have completed these diagram by adding one entry in the bottom right box, in parentheses). ♢

Proposition 3.5. *Let ς be a substitution with seeds. Let Φ be a CA. The following are equivalent:*

(a) *ς is compatible with Φ.*

(b) *There is some ς-seed $\mathbf{s} \in \mathcal{A}^2$ such that $\varsigma^\infty(\mathbf{s}) \in \mathcal{ST}(\Phi)$.*

(c) *For every ς-seed $\mathbf{s} \in \mathcal{A}^2$, we have $\varsigma^\infty(\mathbf{s}) \in \mathcal{ST}(\Phi)$.*

(d) *ς commutes with Φ.*

Proof. See Appendix. □

A substitution ς is *aperiodic* if there is no $\mathsf{z} \in \mathbb{Z} \times \mathbb{N}$ such that $\mathfrak{Sub}(\varsigma) \subseteq \mathsf{Fix}[\sigma^{\mathsf{z}}]$.

Corollary 3.6. *Let $\varsigma : \mathcal{A} \longrightarrow \mathcal{A}^{W \times H}$ be an aperiodic substitution compatible with Φ. Let $\mathbf{A} \in \mathcal{ST}(\Phi)$ be a ς-fixed configuration* [which exists by Prop.3.5(b)], *whose first row is $\mathbf{x} = \mathbf{ab}$, where $\mathbf{a} \in \mathcal{A}^{(-\infty\ldots0]}$ and where $\mathbf{b} \in \mathcal{A}^{[1..\infty)}$ is Φ-periodic* [as in Thm.1.1]*. Then $\overline{\mathcal{O}_\Phi}(\mathbf{x})$ has a p-adic odometer as a factor, for at least one prime factor p of H.*

Proof. See Appendix. □

Let $n \in \mathbb{N}$, and let $\mathcal{A} := \mathbb{Z}_{/n}$. The $\mathbb{Z}_{/n}$*-ratchet CA* is the left-permutative CA $\Psi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ with right-sided local rule $\psi : \mathcal{A}^{\{0,1\}} \longrightarrow \mathcal{A}$ defined

$$\psi(a,b) \quad := \quad \begin{cases} a & \text{if} \quad b \neq n'; \\ a+1 & \text{if} \quad b = n'; \end{cases} \qquad \text{where } n' := n-1.$$

For example, the $\mathbb{Z}_{/2}$-ratchet CA is just the Ledrappier CA shown in Figure 1. Figure 2 shows a spacetime diagram of $\mathbb{Z}_{/4}$-ratchet CA. This diagram is visibly self-similar, and some of the columns are strongly reminiscent of the 4-adic odometer, as explained by the next result.

Proposition 3.7. *Let $n \in \mathbb{N}$, let $\mathcal{A} := \mathbb{Z}_{/n}$, and let $n' := n-1$. Let $\Psi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ be the $\mathbb{Z}_{/n}$-ratchet CA. Then*

(a) *Ψ is compatible with the substitution $\varsigma : \mathcal{A} \longrightarrow \mathcal{A}^{2 \times n}$ defined by*

$$\varsigma(0) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}; \quad \varsigma(1) = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{bmatrix}; \quad \varsigma(2) = \begin{bmatrix} 0 & 2 \\ 0 & 2 \\ \vdots & \vdots \\ 0 & 2 \end{bmatrix}; \quad \ldots \quad \varsigma(n-2) = \begin{bmatrix} 0 & n-2 \\ 0 & n-2 \\ \vdots & \vdots \\ 0 & n-2 \end{bmatrix}; \quad \varsigma(n') = \begin{bmatrix} 0 & n' \\ 1 & n' \\ \vdots & \vdots \\ n' & n' \end{bmatrix};$$

(b) *Let $\mathbf{a} = [\ldots 0,0,0,0,n'.0,0,0,0\ldots]$. Then $\mathbf{ST}_\Psi(\mathbf{a})$ is a ς-fixed point.*

(c) *$(\overline{\mathcal{O}_\Psi}(\mathbf{a}), \Psi)$ is conjugate to a $\mathcal{Z}(n)$-odometer (where $\mathcal{Z}(n)$ is the n-adic integers).*

Proof. **(a,b)** Note that $\mathcal{L}_2(\varsigma) = \mathcal{A}^2$, and that ψ commutes with ς (this can be checked by direct computation, similar to Example 3.4). Also, $[n', 0]$ is a seed for ς, so Proposition 3.5(c) says that the ς-fixed array $\mathbf{S} := \varsigma^\infty([1,0])$ is in $\mathcal{ST}(\Psi)$. But the zeroth row of $\mathbf{S}$ is $\mathbf{a}$; hence $\mathbf{S} = \mathbf{A}$.

(c) Suppose $\mathbf{A} = [A_s^t]_{s \in \mathbb{Z}, t \in \mathbb{N}}$. For all $k \in \mathbb{N}$, let $\mathbf{C}_k := [A^t_{-2^k}]_{t \in \mathbb{N}} \in \mathcal{A}^{\mathbb{N}}$ be the -2^kth column of $\mathbf{A}$. Then $\mathbf{C}_k$ is the sequence of the kth digit in the standard 'n-ary number' representation of the n-ary odometer. That is,

$$\mathbf{C}_k = [\underbrace{0,\ldots,0}_{n^k}, \underbrace{1,\ldots,1}_{n^k}, \underbrace{2,\ldots,2}_{n^k}, \ \ldots\ldots \ \underbrace{n',\ldots,n'}_{n^k}, \underbrace{0,\ldots,0}_{n^k}, \underbrace{1,\ldots,1}_{n^k}, \ \ldots\ldots]$$

This yields an obvious surjection $\Gamma : \overline{\mathcal{O}_\Psi}(\mathbf{a}) \longrightarrow \mathcal{Z}(n)$. Also, Γ is injective, because the information in the columns $\{\mathbf{C}_k\}_{k=0}^\infty$ is sufficient to reconstruct all the other columns in the Ψ-spacetime diagram $\mathbf{A}$. □

If $\mathcal{A} := \mathbb{Z}_{/2}$ and $R \in \mathbb{N}$, then the range R *Coven CA* is the CA $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ with local rule $\phi(x_0, x_1, \ldots, x_R) = x_0 + x_1 x_2 \cdots x_R$; these were introduced in [CH79]. For example, the range 1 Coven CA is just the Ledrappier CA with local rule $\phi(x_0, x_1) = x_0 + x_1$ [Figure 1], while the range 2 Coven CA has local rule $\phi(x_0, x_1, x_2) = x_0 + x_1 x_2$ [Figure 3]. Nonlinear (i.e. $R \geq 2$) Coven CA exhibit self-similar spacetime diagrams which *cannot* be explained simply by compatibility with a substitution map. Also, we remark that the configuration in Figure 3 is *not* automatic (see [vH03] for an introduction to automatic configurations). Instead, these diagrams are self-similar because they can be 'recoded' as the diagrams of

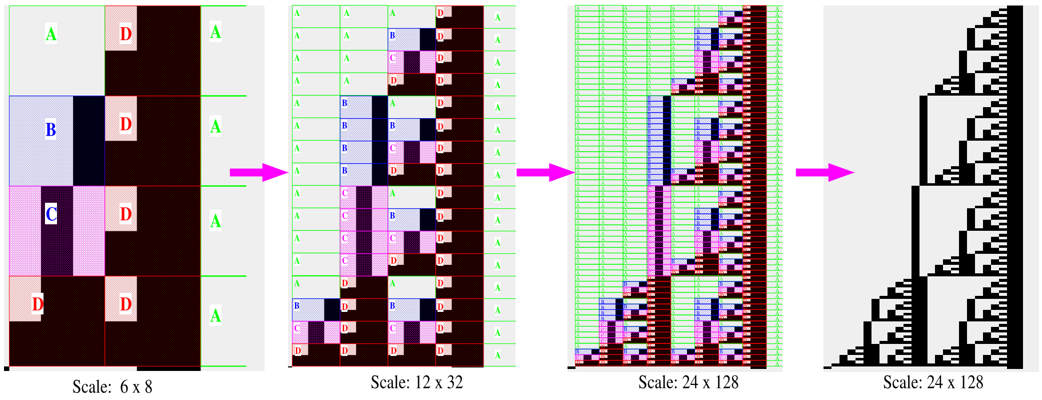


Figure 3: Self-similarity in the Coven CA with local rule $\phi(x_0, x_1, x_2) = x_0 + x_1x_2$. The left three images are the same spacetime diagram, shown on larger and larger scales. The alphabetic labels show how this spacetime diagram can be obtained from Figure 2 via the function Ξ described in Example 3.8.

ratchet CA, which are self-similar by Proposition 3.7. We will explain this in Proposition 3.9, but first we illustrate with an example.

EXAMPLE 3.8: Let $\mathcal{A} := \mathbb{Z}_{/2}$ and let $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ be the range 2 Coven CA with local rule $\phi(x_0, x_1, x_2) = x_0 + x_1x_2$. Let $\mathbf{a} := [0, 0, 0]$, $\mathbf{b} := [0, 0, 1]$, $\mathbf{c} := [0, 1, 0]$ and $\mathbf{d} := [0, 1, 1]$. Let $\mathcal{B} := \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}\} \subset \mathcal{A}^3$, and let $\mathfrak{B} \subset \mathcal{A}^{\mathbb{Z}}$ be the set of all sequences obtained by concatenating words from $\mathcal{B}$, such that a word boundary lies at zero. Clearly, $\mathfrak{B}$ is σ^3-invariant, and it can be checked by direct computation that $\Phi^2(\mathfrak{B}) \subseteq \mathfrak{B}$. Let $\Xi : (\mathbb{Z}_{/4})^{\mathbb{Z}} \longrightarrow \mathfrak{B}$ be the bijection with local rule given by $\xi(0) := \mathbf{a}$, $\xi(1) := \mathbf{b}$, $\xi(2) := \mathbf{c}$, and $\xi(3) := \mathbf{d}$.

Let $\Psi : (\mathbb{Z}_{/4})^{\mathbb{Z}} \longrightarrow (\mathbb{Z}_{/4})^{\mathbb{Z}}$ be the $\mathbb{Z}_{/4}$-ratchet CA. Direct computation shows that $\Xi \circ \sigma = \sigma^3 \circ \Xi$ and $\Xi \circ \Phi^2 = \Psi \circ \Xi$. In other words, Ξ is a dynamical isomorphism from $((\mathbb{Z}_{/4})^{\mathbb{Z}}, \Psi, \sigma)$ to $(\mathfrak{B}, \Phi^2, \sigma^3)$.

The first row of Figure 3 is $[\ldots 0, 0, 0, 0, 1, 1 . 0, 0, 0 \ldots]$, which equals $[\ldots \mathbf{a}, \mathbf{a}, \mathbf{d} . \mathbf{a}, \mathbf{a}, \ldots]$ (an element of $\mathfrak{B}$), which is the Ξ-image of $[\ldots 0, 0, 3 . 0, 0, \ldots]$, which is the first row of Figure 2. Thus, Ξ maps the spacetime diagram of Figure 2 into that of Figure 3. Proposition 3.7(d) implies that Figure 3 is conjugate to a dyadic odometer. $\diamondsuit$

Example 3.8 generalizes as follows.

Proposition 3.9. *Let $\mathcal{A} := \mathbb{Z}_{/2}$ and let $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ be the range R Coven CA. Let $\mathcal{C} := \mathbb{Z}_{/2^R}$ and let $\Psi : \mathcal{C}^{\mathbb{Z}} \longrightarrow \mathcal{C}^{\mathbb{Z}}$ be the $\mathcal{C}$-ratchet CA.*

(a) *There is a (Φ^2, σ^R)-invariant subset $\mathfrak{B} \subset \mathcal{A}^{\mathbb{Z}}$ such that $(\mathfrak{B}, \Phi^2)$ is isomorphic to $(\mathcal{C}, \Psi)$.*

(b) *The point $\mathbf{x} := [\ldots 0, 0, 0, 0, \overbrace{1, \ldots, 1}^{R} . 0, 0, 0, \ldots]$ is in $\mathfrak{B}$, and $(\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi^2)$ is conjugate to $(\overline{\mathcal{O}_\Phi}(\mathbf{a}), \Psi)$, where $\mathbf{a} \in \mathcal{C}^{\mathbb{Z}}$ is as in Proposition* 3.7(b,c)*. Hence $(\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi)$ is isomorphic to a dyadic odometer.*

Proof. **(a)** Let $\mathcal{B} := \{\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{2^R-1}\} \subset \mathcal{A}^{R+1}$, where $\mathbf{b}_0 := [0, \ldots, 0, 0]$, $\mathbf{b}_1 := [0, \ldots, 0, 1]$, $\mathbf{b}_2 := [0, \ldots, 1, 0], \ldots,$ $\mathbf{b}_{2^R-1} := [0, 1, \ldots, 1, 1]$. Let $\mathfrak{B} := \mathcal{B}^{\mathbb{Z}}$, as a subset of $\mathcal{A}^{\mathbb{Z}}$. Clearly $\mathfrak{B}$ is σ^R-invariant. The function $\xi : \mathcal{C} \ni n \mapsto \mathbf{b}_n \in \mathcal{B}$ yields a bijection

$\Xi : \mathcal{C}^{\mathbb{Z}} \longrightarrow \mathfrak{B}$, and direct computation shows that $\Xi \circ \sigma = \sigma^{R+1} \circ \Xi$ and $\Xi \circ \Phi^2 = \Psi \circ \Xi$, so Ξ is a dynamical isomorphism from $(\mathcal{C}^{\mathbb{Z}}, \Psi, \sigma)$ to $(\mathfrak{B}, \Phi^2, \sigma^{R+1})$. It follows that $\mathfrak{B}$ is also Φ^2-invariant.

(b) $\mathbf{x} = \Xi(\mathbf{a})$ where $\mathbf{a}$ is as in Proposition 3.7(b); thus Proposition 3.7(c) states that $(\overline{\mathcal{O}_\Psi}(\mathbf{a}), \Psi)$ is conjugate to the 2^{R+1}-adic (hence dyadic) odometer. Thus, part (a) implies that $(\overline{\mathcal{O}_{\Phi^2}}(\mathbf{x}), \Phi^2)$ is also isomorphic to a dyadic odometer. Thus $(\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi)$ is also isomorphic to a dyadic odometer. □

Acknowledgements. Initial work was done while the authors were on research leave at Wesleyan University, and was partially supported by the Van Vleck Fund. We thank Ethan Coven for generous hospitality and many stimulating discussions. This work was also supported by NSERC Canada.

References

[AvHP$^+$97] J.-P. Allouche, F. von Haeseler, H.-O. Peitgen, A. Petersen, and G. Skordev. Automaticity of double sequences generated by one-dimensional linear cellular automata. *Theoret. Comput. Sci.*, 188(1-2):195–209, 1997.

[BS95] Jorge Buescu and Ian Stewart. Liapunov stability and adding machines. *Ergodic Theory Dynam. Systems*, 15(2):271–290, 1995.

[BvHPS03] A. Barbé, Fritz von Haeseler, Heinz-Otto Peitgen, and Gencho Skordev. Rescaled evolution sets of linear cellular automata on a cylinder. *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, 13(4):815–842, 2003.

[CH79] Ethan M. Coven and G. A. Hedlund. Periods of some nonlinear shift registers. *J. Combin. Theory Ser. A*, 27(2):186–197, 1979.

[CPY07] Ethan M. Coven, Marcus Pivato, and Reem Yassawi. Prevalence of odometers in cellular automata. *Proc. Amer. Math. Soc.*, 135(3):815–821 (electronic), 2007.

[CY07] Ethan M. Coven and Reem Yassawi. Every odometer can be embedded can be embedded in a cellular automaton with local rule $t_0 + t_1$. *(preprint)*, 2007.

[Fog02] N. Pytheas Fogg. *Substitutions in Dynamics, Arithmetics and Combinatorics*. Springer-Verlag, Berlin, 2002.

[Luc78] E. Lucas. Sur les congruences des nombres eulériens et les coefficients différentiels des functions trigonométriques suivant un module premier. *Bull. Soc. Math. France*, 6:49–54, 1878.

[Que87] Martine Queffélec. *Substitution dynamical systems—spectral analysis*, volume 1294 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1987.

[Tak93] Satoshi Takahashi. Cellular automata, fractals and multifractals: space-time patterns and dimension spectra of linear cellular automata. In *Chaos in Australia (Sydney, 1990)*, pages 173–195. World Sci. Publishing, River Edge, NJ, 1993.

[vH03] Fritz von Haeseler. *Automatic sequences*, volume 36 of *de Gruyter Expositions in Mathematics*. Walter de Gruyter & Co., Berlin, 2003.

[vHPS01] Fritz von Haeseler, Heinz-Otto Peitgen, and Gencho Skordev. Self-similar structure of rescaled evolution sets of cellular automata, I and II. *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, 11(4):913–926 and 927–941, 2001.

[Wil87] Stephen J. Willson. 'The equality of fractional dimensions for certain cellular automata' and 'Computing fractal dimensions for additive cellular automata'. *Phys. D*, 24(1-3):179–189 and 190–206, 1987.

Appendix: Proofs

Proof of Theorem 1.2 : The Chinese Remainder Theorem implies that any non-trivial CA Φ with rule $\Phi(\mathbf{x}) = \mathbf{x} + \sum_{i=1}^{r} a_i\sigma^i(\mathbf{x})$ on $(\mathbb{Z}_{/N})^{\mathbb{Z}}$ is topologically conjugate to $\Pi_{j=1}^{n}\Phi_{q_j}$, where Φ_{q_j} acts on $(\mathbb{Z}_{/q_j})^{\mathbb{Z}}$ by $\Phi_{q_j}(\mathbf{x}) := \mathbf{x} + \sum_{i=1}^{r} a_i\sigma^i(\mathbf{x})$. The conditions on the a_i's guarantee that each Φ_{q_j} is non-trivial. Note that $\Pi_{j=1}^{n}(\mathcal{Z}(q_j),\tau)$ is topologically conjugate to $(\mathcal{Z}(Q),\tau)$. We will show that $\Pi_{j=1}^{n}(\mathcal{Z}(q_j),\tau)$ can be embedded in $\Pi_{j=1}^{n}\Phi_{q_j}$.

Case 1. Suppose first that the multiplicity of q in $\mathcal{Q}$ is infinite for each p in $\{q_1, q_2, \dots q_n\}$. Find $\mathbf{x} \in \mathcal{A}^{\mathbb{Z}}$ such that $\mathbf{x}_{[0\dots\infty)}$ is Φ-fixed and such that $\mathcal{O}_\Phi(\mathbf{x}) := \left\{\Phi^t(\mathbf{x})\ ;\ t \in \mathbb{N}\right\}$ is infinite. This can be done, since Φ_{q_j} is conjugate to a full one sided shift, which has fixed points — thus one can find some Φ_{q_j}-fixed $\mathbf{x}_{[0\dots\infty)}$; further if x_{-1} is chosen so that $\mathbf{x}_{[-1\dots\infty)}$ is not fixed, then the proof of Theorem 1.3 shows that $\mathbf{x}$ has an infinite Φ_{q_j} orbit. Using Theorem 4 in [CPY07], Φ_{q_j} embeds $(\mathcal{Z}(q_j),\tau)$.

Case 2. Suppose that $\mathcal{P} = \mathcal{P}_f \cup \mathcal{P}_i$ where p in $\mathcal{P}_f$ have finite multiplicity in $\mathcal{Q}$ and p in $\mathcal{P}_i$ have infinite multiplicity in $\mathcal{Q}$. Let $P := \Pi_{p\in\mathcal{P}_f}p$. As in the Corollary to Theorem 1 in [CPY07], find $\mathbf{x} \in \mathcal{A}^{\mathbb{Z}}$ such that $\mathbf{x}_{[0\dots\infty)}$ is Φ_{q_1}-periodic with least period P, and such that $\mathcal{O}_\Phi(\mathbf{x})$ is infinite. Then Φ_{q_1} embeds $(\mathcal{Z}(P, q_1, q_1, \dots),\tau)$, and, by Case 1, Φ_{q_j} embeds $(\mathcal{Z}(q_j),\tau)$ for $1 < j \le n$. The embedding result follows. That no other odometer can be embedded in these linear CA is proved similarly to the result for $\Phi(x) = x + \sigma(x)$ in [CY07]. □

Lemma 3.10. *Let $\mathcal{A} := \mathbb{Z}_{/2}$, and suppose that $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ and $\mathbf{x} = (x_i)_{i=-\infty}^{\infty} \in \mathcal{A}^{\mathbb{Z}}$ satisfy the conditions of* Theorem 1.3. *Then for each $j \ge 1$ and each $k \ge 0$, we have*

$$\sum_{p=1}^{L} x_{2^k(a_p-a_1)+a_1-1+j} = 0\,.$$

Thus $\Phi^{2^k}(\mathbf{x})|_{-(2^k a_1-a_1+1)+j} = x_{-(2^k a_1-a_1+1)+j}$ for each $k \ge 0$ and $j \ge 1$.

Proof. We prove this by induction on k. Since $\mathbf{x}_{[0\dots\infty)}$ is fixed by Φ, we have $x_k + \sum_{p=1}^{L} x_{k+a_p} = x_k$, for each $k \ge 0$, so that $\sum_{p=1}^{L} x_{a_p-1+j} = 0$ is true for each $j \ge 1$. Let $\Delta_i := a_{i+1} - a_i$, for $1 \le i \le L-1$, and assume that for each $j \ge 1$,

$$\sum_{p=1}^{L} x_{2^k(a_p-a_1)+a_1-1+j} = \sum_{p=1}^{L} x_{a_1-1+j+2^k(\sum_{i=1}^{p-1}\Delta_i)} = 0. \tag{3.3}$$

Given a positive j, let $\star_p := 2^{k+1}(a_p - a_1) + a_1 - 1 + j$ for $1 \le p \le L$, and define $*_{11} := \star_1$ and $*_{1p} := \star_1 + 2^k\left(\sum_{i=1}^{p-1}\Delta_i\right)$ for $p = 2, \dots L$. By Equation 3.3, $\sum_{p=1}^{L} x_{*_{1p}} = 0$. For $2 \le,\ q \le L$, define $*_{q1} := *_{1q}$, and $*_{qp} := *_{q1} + 2^k\left(\sum_{i=1}^{p-1}\Delta_i\right)$. Using Equation 3.3 and $j_q := j + 2^k\left(\sum_{i=1}^{q-1}\Delta_i\right)$, we have $\sum_{p=1}^{L} x_{*_{qp}} = 0$.

Consider the matrix $\{x_{*_{pq}}\}_{p,q=1}^{L}$. We have the following two claims:

Claim 1: $*_{pq} = *_{qp}$ *for $1 \le p, q \le L$.*

Proof. If $q=1$ or $p=1$ this is true by definition. If $2 \leq p,q \leq L$, $*_{pq} = *_{1p} + 2^k \sum_{i=1}^{q-1} \Delta_i = \star_1 + 2^k \sum_{i=1}^{p-1} \Delta_i + 2^k \sum_{i=1}^{q-1} \Delta_i = *_{1q} + 2^k \sum_{i=1}^{p-1} \Delta_i = *_{qp}$. ◇ Claim 1

Claim 2: $*_{pp} = \star_p$.

Proof. $*_{pp} = *_{1p} + 2^k \left(\sum_{i=1}^{p-1} \Delta_i \right) = \star_1 + 2^{k+1} \left(\sum_{i=1}^{p-1} \Delta_i \right) = a_1 - 1 + j + 2^{k+1}(a_p - a_1) = \star_p$. ◇ Claim 2

These last two claims tell us that the matrix $\{x_{*_{pq}}\}_{p,q=1}^{L}$ is a symmetric $0,1$-matrix each of whose rows sum to zero. Thus $0 = \sum_{p=1}^{L} \sum_{q=1}^{L} x_{*_{pq}} = \sum_{p=1}^{L} x_{*_{pp}} = \sum_{p=1}^{L} x_{\star_p}$, which shows that $\sum_{p=1}^{L} x_{2^{k+1}(a_p - a_1) + a_1 - 1 + j} = 0$.

□

Lemma 3.11. *Let $\mathcal{A} := \mathbb{Z}_{/2}$, and suppose that $\Phi : \mathcal{A}^{\mathbb{Z}} \longrightarrow \mathcal{A}^{\mathbb{Z}}$ and $\mathbf{x} = (x_i)_{i=-\infty}^{\infty} \in \mathcal{A}^{\mathbb{Z}}$ satisfy the conditions of* Theorem 1.3. *Then for each $k \geq 0$, $\Phi^{2^k}(\mathbf{x})|_{-(2^k a_1 - a_1 + 1)} \neq x_{-(2^k a_1 - a_1 + 1)}$.*

Proof. We prove this by induction on k. If $\mathbf{x}$ satisfies the conditions of Theorem 1.3, then $\Phi(\mathbf{x})|_{-1} \neq x_{-1}$. Next, assume that $\Phi^{2^k}(\mathbf{x})|_{-(2^k a_1 - a_1 + 1)} \neq x_{-(2^k a_1 - a_1 + 1)}$. Thus

$$x_{-(2^k a_1 - a_1 + 1)} + x_{a_1 - 1} + \sum_{p=2}^{L} x_{a_1 - 1 + 2^k(a_p - a_1)} \neq x_{-(2^k a_1 - a_1 + 1)}, \tag{3.4}$$

What follows is essentially the same as that of the previous lemma, except that this time we have a symmetric, 0-1 matrix all of whose rows sum to 0, *except* the first, which sums to one. We claim that

$$x_{a_1 - 1} + \sum_{p=2}^{L} x_{a_1 - 1 + 2^{k+1}(a_p - a_1)} = 1\,. \tag{3.5}$$

Let $*_{1p} := a_1 - 1 + 2^k(a_p - a_1)$, for $1 \leq p \leq L$. For $2 \leq p \leq L$, let $*_{p1} := *_{1p} = a_1 - 1 + 2^k(a_p - a_1)$, and $*_{pq} := a_1 - 1 + 2^k(a_p - a_1) + 2^k(a_q - a_1)$. Let $j_p := 2^k(a_p - a_1)$. Lemma 3.10 tells us that $x_{a_1 - 1 + j} + \sum_{p=2}^{L} x_{2^k(a_p - a_1) + a_1 - 1 + j} = 0$, so $\sum_{q=1}^{L} x_{*_{pq}} = 0$; Equation 3.4 implies that $\sum_{q=1}^{L} x_{*_{1q}} = 1$.

As in Lemma 3.10,we have a symmetric 0-1 matrix $[x_{*_{pq}}]_{p,q=1}^{L}$ whose diagonal terms are the summands in Equation 3.5, and all of whose rows sum to zero, save the first row. The result follows. □

Proof of Theorem 1.3. Lemma 3.11 tells us that $k_n \geq 2^n a_1 - a_1 + 1$, and Lemma 3.10 tells us that $k_n = 2^n a_1 - a_1 + 1$. □

Proof of Lemma 3.2. **(a)** is a standard argument [Fog02, §1.2.6]. For **(b)** note that $\mathbf{s} \in \mathcal{L}_2(\varsigma)$. For any $n \in \mathbb{N}$, let $\mathcal{R}_n := \left\{ \mathbf{r} \in \mathcal{A}^2 \; ; \; \mathbf{r} \text{ occurs in } \boldsymbol{\varsigma}^n(\mathbf{s}) \right\}$. Then $\mathcal{R}_1 \subseteq \mathcal{R}_2 \subseteq \mathcal{R}_3 \subseteq \cdots \subseteq \mathcal{L}_2(\varsigma)$. Let $\mathcal{R}_\infty = \bigcup_{n=1}^{\infty} \mathcal{R}_n$. Then $\mathcal{R}_\infty = \mathcal{L}_2(\varsigma)$, because $\mathbf{s}$ is a ς-seed. But $\mathcal{R}_1 \subseteq \mathcal{R}_2 \subseteq \cdots \subseteq \mathcal{R}_\infty$ are finite sets, so there exists $n \in \mathbb{N}$ such that $\mathcal{R}_\infty = \mathcal{R}_n$. □

Proof of Proposition 3.5. "(**a**) $\Rightarrow$ (**c**)" If $\mathbf{s}$ is a ς-seed, then $\boldsymbol{\varsigma}^\infty(\mathbf{s}) \in \mathfrak{Sub}(\varsigma)$. If $\mathfrak{Sub}(\varsigma) \subseteq \mathcal{ST}(\Phi)$, then $\boldsymbol{\varsigma}^\infty(\mathbf{s}) \in \mathcal{ST}(\Phi)$.

"(**c**) $\Rightarrow$ (**b**)" is immediate. For "(**b**) $\Rightarrow$ (**a**)", let $\mathbf{S} = \boldsymbol{\varsigma}^\infty(\mathbf{s})$, and suppose $\mathbf{S} \in \mathcal{ST}(\Phi)$. Then the σ-orbit closure of $\mathbf{S}$ is contained in $\mathcal{ST}(\Phi)$, because $\mathcal{ST}(\Phi)$ is closed and σ-invariant. Thus, Lemma 3.2(a) says $\mathfrak{Sub}(\varsigma) \subseteq \mathcal{ST}(\Phi)$; ie. ς is compatible with ϕ.

"(**d**) $\Rightarrow$ (**c**)" Let $\mathbf{s} = [a, b]$ be a ς-seed and let $\mathbf{S} := \boldsymbol{\varsigma}^\infty(\mathbf{s}) = \overline{\mathbf{A}|\mathbf{B}}$, where $\mathbf{A} = \boldsymbol{\varsigma}^\infty(a)$ and $\mathbf{B} := \boldsymbol{\varsigma}^\infty(b)$.

Claim 1: *For all $n \in \mathbb{N}$, let $\mathbf{A}_n := \boldsymbol{\varsigma}^n(a)$ and $\mathbf{B}_n := \boldsymbol{\varsigma}^n(b)$. Then $\overline{\mathbf{A}_n|\mathbf{B}_n}$ is $\mathcal{ST}(\Phi)$-admissible.*

Proof. Case (n=1): By definition, $[a, b] \in \mathcal{L}_2(\varsigma)$, and by hypothesis, ϕ commutes with ς, so $\overline{\mathbf{A}_1|\mathbf{B}_1} = \varsigma[a, b]$ is $\mathcal{ST}(\Phi)$-admissible.

Induction: Suppose $\overline{\mathbf{A}_n|\mathbf{B}_n}$ is $\mathcal{ST}(\Phi)$-admissible. Let $\left[\begin{smallmatrix} x & y \\ z & \end{smallmatrix}\right]$ be a triomino appearing somewhere in $\overline{\mathbf{A}_{n+1}|\mathbf{B}_{n+1}}$; we must show that $\left[\begin{smallmatrix} x & y \\ z & \end{smallmatrix}\right]$ is $\mathcal{ST}(\Phi)$-admissible [ie. that $z = \phi(x, y)$]. Now, $\overline{\mathbf{A}_{n+1}|\mathbf{B}_{n+1}} = \boldsymbol{\varsigma}(\overline{\mathbf{A}_n|\mathbf{B}_n})$, so there is some triomino $\left[\begin{smallmatrix} u & v \\ w & \end{smallmatrix}\right]$ in $\overline{\mathbf{A}_n|\mathbf{B}_n}$ such that $\left[\begin{smallmatrix} x & y \\ z & \end{smallmatrix}\right]$ appears inside $\boldsymbol{\varsigma}\left[\begin{smallmatrix} u & v \\ w & \end{smallmatrix}\right]$. By definition, $[u, v] \in \mathcal{L}_2(\varsigma)$, and by hypothesis, ϕ commutes with ς. Hence $\boldsymbol{\varsigma}\left[\begin{smallmatrix} u & v \\ w & \end{smallmatrix}\right]$ is a $\mathcal{ST}(\Phi)$-admissible fragment, which in particular means that $\left[\begin{smallmatrix} x & y \\ z & \end{smallmatrix}\right]$ is $\mathcal{ST}(\Phi)$-admissible.

This works for any $\left[\begin{smallmatrix} x & y \\ z & \end{smallmatrix}\right]$ in $\overline{\mathbf{A}_{n+1}|\mathbf{B}_{n+1}}$. Thus, $\overline{\mathbf{A}_{n+1}|\mathbf{B}_{n+1}}$ is $\mathcal{ST}(\Phi)$-admissible, because $\mathcal{ST}(\Phi)$ is the SFT generated by the set of triominos in eqn.(3.1). ◇ `Claim 1`

It follows that $\mathbf{S} = \overline{\mathbf{A}|\mathbf{B}}$ is $\mathcal{ST}(\Phi)$-admissible.

"(**b**) $\Rightarrow$ (**d**)" Suppose $\mathbf{S} = \boldsymbol{\varsigma}^\infty(\mathbf{s})$ for some ς-seed $\mathbf{s} \in \mathcal{A}^2$. If $\mathbf{S} \in \mathcal{ST}(\Phi)$, then ς commutes with Φ on all $[u, v]$ which occur in $\mathbf{S}$. But Lemma 3.2(b) says this is all of $\mathcal{L}_2(\varsigma)$. □

Proof of Corollary 3.6. $\mathbf{x}$ has infinite Φ-orbit because otherwise, $\mathbf{A}$ would be fixed under some vertical shift, contradicting the aperiodicity of ς. Thus, Theorem 1.1 says $\mathfrak{O} := (\overline{\mathcal{O}_\Phi}(\mathbf{x}), \Phi)$ is isomorphic to some odometer. We must show that $\mathfrak{O}$ has a p-adic odometer as a factor, for some prime p dividing H.

For all $k \in \mathbb{N}$, let $\mathbf{A}_k := \mathbf{A}_{(-W^k \ldots 0] \times \mathbb{N}} = [\Phi^t(\mathbf{x})_{(-W^k \ldots 0]}]_{t=0}^\infty$ (i.e. the first W^k 'columns' in the spacetime diagram of $\mathbf{x}$). Each $\mathbf{A}_k$ is vertically periodic (because $\mathfrak{O}$ is an odometer); let T_k be its minimal period. Thus, $T_0 \leq T_1 \leq T_2 \leq \cdots$.

Claim 2: (a) *T_k divides $H^k T_0$.* (b) $\lim\limits_{k\to\infty} T_k = \infty$.

Proof. **(a)** $\boldsymbol{\varsigma}(\mathbf{A}) = \mathbf{A}$, so $\boldsymbol{\varsigma}(\mathbf{A}_{k-1}) = \mathbf{A}_k$, so $\mathbf{A}_k$ is vertically (HT_{k-1})-periodic, so its *least* period T_k must divide HT_{k-1}. By induction, this means T_k divides $H^k T_0$.

(b) By contradiction, suppose the sequence $\{T_k\}_{k=1}^\infty$ was bounded. Then there would be some k such that $T_k = T_{k+1} = T_{k+2} = \cdots$, and then $\mathbf{x}$ would be Φ^{T_k}-periodic, contradicting the fact that $\mathbf{x}$ has infinite Φ-orbit. ◇ `Claim 2`

For any N, Claim 1(b) yields some $k \geq N$ such that $T_k \geq H^N T_0$. Let $d := \gcd(T_k, T_0)$, and let $T'_k = T_k/d$ and $T'_0 := T_0/d$; then $T'_k \geq H^N T'_0 \geq H^N$. But Claim 1(a) says that T_k divides $H^k T_0$, which means T'_k divides $H^k T'_0$, which means T'_k divides H^k (because T'_k is coprime to T'_0). Thus, all prime factors of T_k are prime factors of H. But $T'_k \geq H^N$, so T_k must be divisible by p^N for at least one prime factor p of H. It follows that p^N divides T_k, which means that $\mathfrak{O}$ contains a factor of minimal period p^N. But N can be made arbitrarily large, so $\mathfrak{O}$ must have a p-adic odometer as a factor. □

Journées Automates Cellulaires 2008 (Uzès), pp. 130-140

TRANSLATING PARTITIONED CELLULAR AUTOMATA INTO CLASSICAL TYPE CELLULAR AUTOMATA

VICTOR POUPET

Laboratoire d'Informatique Fondamentale (LIF), UMR 6166 CNRS, Université de Provence
CMI, 39 rue Joliot-Curie, 13453 Marseille Cedex 13
E-mail address: victor.poupet@lif.univ-mrs.fr
URL: http://www.lif.univ-mrs.fr/ vpoupet/

Abstract. Partitioned cellular automata are a variant of cellular automata that was defined in order to make it very simple to create complex automata having strong properties such as number conservation and reversibility (which are often difficult to obtain on cellular automata). In this article we show how a partitioned cellular automaton can be translated into a regular cellular automaton in such a way that these properties are conserved.

1. Introduction

A number-conserving cellular automaton is a cellular automaton such that all states of cells are represented by integers and that the sum of all states in any finite configuration (the quiescent state corresponds to the value 0) is unchanged after one transition of the automaton. Number-conservation can be seen as a modelization of the physical law of conservation of mass and energy. Thus number conserving cellular automata can be used to model complex physical phenomena, like fluid dynamics [1] or highway traffic flow [6].

However, it appears that designing number-conserving cellular automata with complex transition rules is very difficult. To solve this problem a new kind of cellular automata has been studied by Morita et al. [3] that use a *partitioned* space. It is then possible to construct complex two-dimensional number-conserving (and reversible) PCA, like for example some that simulate any reversible two-counter machine [4].

Although in a number-conserving partitioned cellular automaton the total weight of a configuration is conserved after each transition, each cell has multiple parts and cannot therefore be regarded as a usual type of CA. In this paper we will show how it is possible to translate any number-conserving partitioned cellular automaton into a classical type number-conserving cellular automaton and thus prove the equivalence of the two models.

Key words and phrases: Partitioned cellular automata, number-conservation, reversibility.

1.1. Definitions

Definition 1.1 (Cellular Automaton)**.** A *cellular automaton* (CA) is a quadruple $\mathcal{A} = (d, \mathcal{Q}, V, f)$ where

- $d \in \mathbb{N}$ is the dimension of the automaton;
- $\mathcal{Q}$ is a finite set called *set of states*;
- $V = \{v_1, \ldots, V_{|V|}\} \subseteq \mathbb{Z}^d$ is a finite set called *neighborhood*;
- $f : \mathcal{Q}^{|V|} \to \mathcal{Q}$ is the *local transition function.*

For a given automaton $\mathcal{A}$, we call *configuration* of $\mathcal{A}$ any function $\mathfrak{C}$ from $\mathbb{Z}^d$ into $\mathcal{Q}$. The set of all configurations is therefore $\mathcal{Q}^{\mathbb{Z}^2}$. From the local function f we can define a global function F

$$\begin{array}{rcccl} F & : & \mathcal{Q}^{\mathbb{Z}^d} & \to & \mathcal{Q}^{\mathbb{Z}^d} \\ & & \mathfrak{C} & \mapsto & \mathfrak{C}' \quad | \quad \forall x \in \mathbb{Z}^d, \mathfrak{C}'(x) = f(\mathfrak{C}(x+v_1), \ldots, \mathfrak{C}(x+v_{|V|})) \end{array}$$

Elements of $\mathbb{Z}^d$ are called *cells.* Given a configuration $\mathfrak{C}$, we will say that a cell c is in state q if $\mathfrak{C}(c) = q$. If we distinguish a *quiescent* state q_0 such that $f(q_0, \ldots, q_0) = q_0$, we will call *finite* configuration any configuration for which only a finite number of cells is not in the quiescent state. If $\mathfrak{C}$ is a finite configuration, so is $F(\mathfrak{C})$.

Cellular automata will be seen as dynamical systems. If the CA is in the configuration $\mathfrak{C}$ at some time, we will say that it is in the configuration $F(\mathfrak{C})$ at the next time. We can therefore define the *evolution* of a CA from a configuration. This evolution is completely determined by $\mathfrak{C}$.

Remark: In the following, we will only consider two-dimensional CA ($d = 2$).

Definition 1.2 (Number-Conservation)**.** A CA will be said to be *number-conserving* if its states are distinct natural numbers ($\mathcal{Q} \subseteq \mathbb{N}$), the state 0 is quiescent, and for every finite configuration $\mathfrak{C}$ the total weight of $\mathfrak{C}$ (sum of all states) is equal to that of $F(\mathfrak{C})$.

Definition 1.3 (Partitioned Cellular Automaton)**.** A two-dimensional partitioned cellular automaton (PCA) is a four-neighbor two-dimensional CA whose cells are divided into four parts : upper, left, lower and right. The next state of each cell is only determined by the current states of the upper partition of the lower cell, the right partition of the left cell, the lower partition of the upper cell and the left partition of the right cell.

Let us denote by $\mathcal{Q}_p$ the set of all states that a partition of a cell can be into. Then there are $\mathcal{Q}_t = |\mathcal{Q}_p|^4$ different states for each cell. However, the local rules of the automaton can be seen as a function of $\mathcal{Q}_p^4 \to \mathcal{Q}_p^4$. As illustrated by Figure 1. $\mathcal{Q}_p$ will be called the *set of partitioned states* of the PCA.

It is very easy to see that for PCA global reversibility is equivalent to local reversibility. It is therefore very easy to design reversible automata using this notion.

Note. There are other kinds of PCA. We could for example divide each cell into 5 parts by adding a central partition. It is also possible to consider PCA in other dimensions. However in this paper we will only work with two-dimensional 4-partitioned PCA.

If we want to consider number-conservation for PCA we have to redefine it. In this case, only the partitioned states will be integers. The total weight of a cell is the sum of the four states in its partitions.

The weight of a finite configuration is the sum of all weights of its cells (only a finite number of cells are in a positive state). The automaton is said to be globally number-conservative if for all finite configuration its total weight is conserved after one transition and

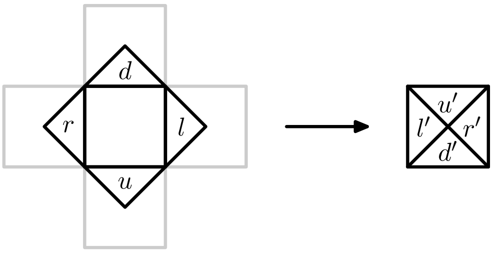


Figure 1: Illustration of the rule $[d, l, u, r] \to [u', r', d', l']$.

locally number-conservative if for each rule $(d, l, u, r) \to (u', r', d', l')$ we have $d + l + u + r = u' + r' + d' + l'$.

It is very easy to see that both local and global number-conservation are equivalent for a PCA and hence it is very easy to design a number conservative PCA even when we want to have it perform a complex task.

1.2. Why Translating into Classical Type CA?

We have seen that the notion of PCA is very useful when working with number-conserving or reversible automata as they are very easy to design and that checking both number-conservation and reversibility can be done by a local study of each rule (therefore very simply and quickly).

However the definition of PCA slightly differs from classical CA. Of course it is possible to consider that a PCA is a classical CA and that it has Q_p^4 different states. However, by doing so we see that what we called a number-conserving PCA is not necessarily a number conserving CA because we gave the same weight to different states (for example the states $(1, 0, 0, 0)$ and $(0, 0, 0, 1)$ have the same weight). In other words, even if the PCA are a subclass of CA, the definition we gave of number-conservation in this specific class (which is the definition that makes it easy to design number conserving PCA) is not the same as the usual number-conservation.

Here we will show how a NCPCA can be translated into a classical NCCA without increasing the number of states.

2. Main Theorem

The rest of the article will be devoted to proving the following theorem:

Theorem 2.1. *Given a PCA $\mathcal{A}_p$ of partitioned states $\mathcal{Q}$, there exists a CA $\mathcal{A}_r$ of states $\mathcal{Q}$ working on the neighborhood V_m^{10} such that $\mathcal{A}_r$ mimics the behavior of $\mathcal{A}_p$ (in a very natural sense that we will explain later). Moreover, if ACA_p is number conserving or reversible (or both) then so is $\mathcal{A}_r$.*

In the following, we will consider a PCA of partitioned states $\mathcal{Q}$ and describe how the regular CA described in the theorem works.

3. Preliminaries

3.1. Main Idea

First, we will explain how to convert configurations of $\mathcal{A}_p$ into configurations of $\mathcal{A}_r$ in such a way that no information is lost so that from such an image configuration the CA $\mathcal{A}_r$ can easily mimic the behavior of $\mathcal{A}_p$. Then we will show how we can ensure that $\mathcal{A}_r$ has the same properties (number-conservation and reversibility) than $\mathcal{A}_p$. These properties will be trivially conserved on valid configurations (configurations that are images of configurations of $\mathcal{A}_p$ by the transformation mentioned above) but not necessarily on invalid ones. In this case we will show that the CA $\mathcal{A}_r$ can detect the irregularities and "freeze" its behavior so that nothing happens that can break number-conservation or reversibility: after all, the identity CA is both conservative and reversible.

3.2. Vocabulary

To avoid confusions between the original PCA and the new CA we will clearly distinguish the vocabulary between the two.

From now on we will call *square cell* a cell of the original PCA. A partition will be one portion of a square cell (a square cell has 4 partitions which are naturally the upper, right, lower and left partitions).

Most of the time, the word *cell* will refer to a cell of the new CA. This automaton works as classical CA do, but on a Moore neighborhood of radius 10 (we will see later why such a neighborhood is needed).

We will call *states* the elements of $\mathcal{Q}$ (the partitioned states of $\mathcal{A}_p$ and the states of the cells in $\mathcal{A}_r$).

The state 0 will be called *blank* state. All other states will be *colored* states. The blank state is assumed to be quiescent. A blank cell is a cell whose state is 0, and a colored cell is a cell whose state is not 0.

If we consider a given partition p in the PCA, there are some partitions that play a particular role from its point of view. The first of these particular partitions is the one that is in front of it (the lower partition of the upper square cell in the case of an upper partition for example). This partition will be called p's *facing* partition. The three other partitions in the same square cell as p will be called p's *brother* partitions.

Later on, we will establish a parallelism between certain patterns of the new CA (called *blocks*) and the partitions of the original one, which will lead to the use of the terms *facing blocks* and *brother blocks* whose meaning will be straightforward.

3.3. Switch and Mix: the Key to Understanding PCA

There is an important property in the way PCA local rules are defined. As shown by Figure 1, the transition is such that four partitions (the *outter* ones shown on the left part of the figure) entirely define the next states in four other partitions (the *inner* ones). But there is more than meets the eye...

Indeed, transitions of a PCA can be split in two *virtual* steps that are purely local transformations: the *switch* and the *mix*. During the switch, all partitions exchange their state with their facing partition. Once this is done, all brother partitions in a square cell "mix" their states to produce the result of the local transition rule.

These two steps are virtual because they happen both in one single step of the automaton. It might seem useless to consider an intermediate step, but it gives the automaton an important property: pure locality. Both steps can now be expressed as local transformations of some parts of the configuration: the switch transforms the pairs of facing partitions whereas the mix transforms the four brother cells inside of a square cell, and for both of these transformations the *affecting* area is the same as the *affected* area[1].

This property is reminiscent of the kind of cellular automata considered by N. Margolus to build small Turing-universal machines [2].

3.4. From the Configurations of $\mathcal{A}_p$ to Those of $\mathcal{A}_r$

Starting from a configuration $\mathfrak{C}$ of $\mathcal{A}_p$, we obtain the configuration $\tau(\mathfrak{C})$ of $\mathcal{A}_r$ by transforming each square-cell into a 4×4 pattern of cells of $\mathcal{A}_r$ as illustrated by Figures 2 and 3.

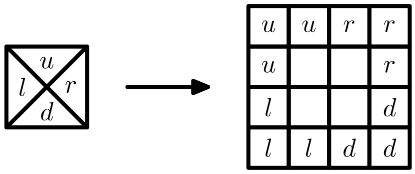


Figure 2: How to translate a square cell into a portion of configuration of $\mathcal{A}_r$.

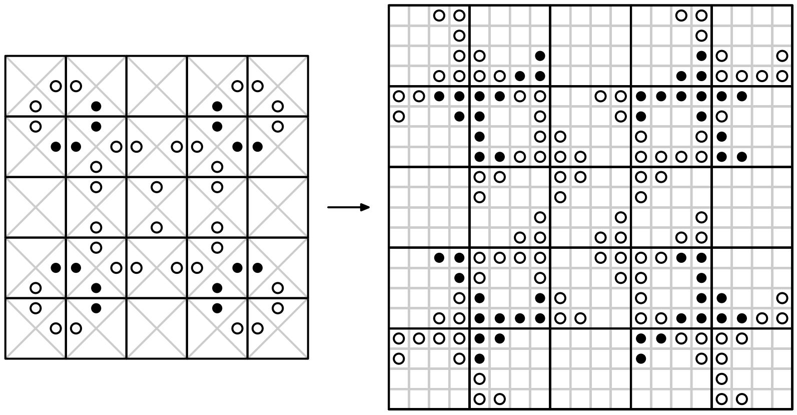


Figure 3: Example of conversion of a configuration of a PCA into a configuration of our new automaton. The original PCA has 3 different states : white, black and blank.

We will call *valid* a configuration of $\mathcal{A}_r$ that is the image by τ of some configuration of $\mathcal{A}_p$ and *invalid* a configuration that is not.

[1]This property is reminiscent of the kind of cellular automata considered by N. Margolus to build small Turing-universal machines [2]. In fact PCA are highly related to the Margolus model as one kind can easily be translated into the other by simple geometric operations on the configurations and local transition rules.

4. Evolution of $\mathcal{A}_r$

The evolution of a cell in $\mathcal{A}_r$ will happen in 3 steps. The cell will always start assuming that it is in a valid configuration and try to apply the rule of $\mathcal{A}_p$ on this valid configuration. However, if it finds that it is not in a valid configuration it will stop its behavior to make sure that no irreversible or non-conserving action is made (the behavior of $\mathcal{A}_p$ needs not be mimicked if the configuration is invalid).

During the first step the cell will try to find the block in which it lies and check that this block is correctly formed. Then, as it will know the orientation of its block, the second step is to look at its facing block and check that it is also well formed. If it is then it will apply the *switch* step with its facing block. The third and last step is then to look at its brother blocks and check that they are well formed and that they have made the switch step. If everything is correct, the blocks will apply the *mix* step, which completes the transition.

All of these 3 steps will in fact take place in one single transition of the CA.

4.1. Finding the Block

A valid configuration of $\mathcal{A}_r$ looks like what is shown in Figure 4.

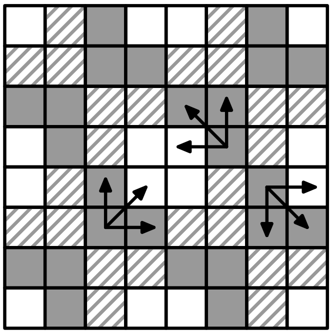


Figure 4: A valid configuration.

In such a configuration, a colored cell can easily determine its block by looking at its immediate neighborhood: if for each of the four quadrants (up-right, up-left, down-left and down-right) it looks at its 3 corresponding neighbors (up, up-right and right for the up-right quadrant for example), there should be exactly one quadrant in which it has exactly one blank neighbor and two colored neighbors in the same state as itself (the correct quadrant is shown for some cells in the figure). When the block is found, the orientation of the block gives the position of the partition that it corresponds to in the pre-image of the configuration (upper, left, lower or right). From there the cell knows where its facing block and its brother blocks should be.

By looking one step farther (radius 2), the cell can determine if the other cells on its block can determine correctly the block they are in (and come to the same conclusion as itself). The block is considered *well formed* if there was exactly one possible block, that the cells in this block all have the same state and that all cells in the block consider themselves as part of this block.

If the considered cell is blank, the situation is slightly more complex. A blank cell can be either a cell of a blank block or a *marking blank* cell (one of the blank cells between the blocks). To find its situation, the cell will look at a Moore neighborhood of radius 5 around itself. In this neighborhood, it will try to identify the well formed colored blocks (as defined previously). According to what it finds, it will either conclude that it is in an invalid configuration (if the blocks are not well formed or do not match the ones with the others) or in a valid one and therefore know its own situation.

The only things that the cell needs to identify are its facing and brother blocks. These are all located in a neighborhood of radius 3. Since a neighborhood of radius 2 is necessary for a colored cell to know if its block is well formed, the blank cell needs a radius 5 neighborhood to determine a possible match for its facing and brother blocks.

Only if the blank cell has managed to find its block (and therefore the location of its facing and brother blocks) does it proceed to the next step (otherwise it remains blank).

Note that it is possible that the blank cell sees only blank cells on its radius 3 neighborhood and therefore cannot find any well formed block by looking at its radius 5 neighborhood. In this case it will not know its position but this is not a problem since, if it is in a well formed configuration, all its facing and brother blocks are blank. Since the blank state is quiescent, the cell should stay blank anyway.

All in all, any cell can find its block and make sure that the other cells in its block consider themselves as being part of the same well formed block by looking at a radius 5 neighborhood around itself.

4.2. Switch

The switch happens between a block and its facing block. During the switch a cell of one block takes the state of its facing block. To make sure that the switch is conservative and reversible we have to make sure that both blocks are well formed and that each considers the other as its facing block.

For a given cell, all the cells in its facing block are located in a radius 3 neighborhood. Since a radius 5 neighborhood was needed to find the block of a cell, by looking at a radius 8 neighborhood, a cell can check that the blocks of all the cells it considers as its facing block also consider themselves as being in its facing block.

If this is the case it take the state of these cells (switch). Otherwise, it does nothing.

The switch is a virtual step, which means that the cell does not actually change its state yet. But if the conditions for the switch are met, and that the next step (mix) does not happen, the cell will really take the switched state at the end of the transition.

4.3. Mix

The mix is very similar to the switch. However it is necessary to check a larger area: the cell must check that all its brother blocks are well formed and that all the facing blocks of its brother blocks are too in order to make sure that all the brother blocks have made a successful switch.

The cells in the facing block of a brother block of a given cell are all located in a radius 5 neighborhood of the cell, which means that by looking at a radius 10 Moore neighborhood around itself the cell can check that all its brother blocks have successfully performed their switch and will also perform the mixing step, meaning that it will look at the states of the facing blocks of its brother blocks and will apply the rule of the PCA to determine its own new state.

If an error is spotted while checking before the mix, the cell does not mix with its brother blocks and therefore keeps the switched state. Otherwise, the switched state is mixed and the switch remains a virtual step.

5. Correctness of the Automaton

5.1. Number-Conservation

The number-conservation is guaranteed by the strong verifications that we do all the time. In fact, in most cases the cell keeps its own state.

First of all it is important to notice that modifications of states occur in block transitions, that is to say that a cell only decides to change its state when it knows that it is part of a well formed block and by the definition of a well formed block all cells of the same block consider themselves in the same block (this means that if a cell c considers that c' is in its block then c' considers that c is in its block). The consequence of this is that each time that c will change its state, this change is made as a "block change" and therefore c' will change its state the same way.

This means that a cell that isn't part of a well formed block will never change its state.

Once we have this in mind, we see that a block will only change its state in two occasions : either after a switch with its facing block or after a mix with its brother blocks.

In the first case the block first checks that the facing block is correctly oriented so that this facing block will in turn take the current block's information. There is therefore a real block exchange, no state is lost nor created, there is evidently number-conservation when a switch occurs.

In the second case the block also checks that all three other brother blocks are well formed and correctly oriented and that they have already performed a switch so that they will in turn decide to perform the mix. This ensures that all four apply the rule of the original PCA and thus as the rules of the PCA are number-conserving (this is our hypothesis) the mix operation is also number-conserving.

Therefore, every time there is any kind of change in the states of a block we have guaranteed that this change is compensated by the change of 1 or 3 other blocks (and of course this guarantees the conservation at the level of the cells because every block has exactly 3 cells that have all the same state).

5.2. Reversibility

If $\mathcal{A}_p$ is reversible, then so is $\mathcal{A}_r$: given a configuration $\mathfrak{C}$ of $\mathcal{A}_r$, we can rebuild its predecessor (because it is reversible, the CA is bijective as a consequence of the Moore-Myhill theorem).

The important thing to see is that a well formed block stays well formed as $\mathcal{A}_r$ evolves and that a non well formed block cannot become well formed (with the exception of large areas of blank cells that can be considered as both well formed and not well formed, without any consequence because the blank state is quiescent).

Because of this, one can check the status of a given cell. If it is in no well formed block or that its facing block is not well formed we know that it didn't change its state. If it is part of a well formed block, that its facing block is also well formed but that one of its brother blocks or the facing block of one of its brother blocks is not well formed, then the block has simply switched with its facing block. Lastly, if all the surrounding blocks are correct, the cell (and its whole block) has applied the rule of the reversible PCA $/ACA_p$, which is a reversible local rule, so the pre-image of the cell can be determined by looking at its brother blocks' states.

5.3. Simulation of the Original PCA

In order to be useful in any way, our automaton must be able to simulate the behaviour of the original PCA.

This simulation is very easy to realize as the rules of our automaton have been designed for it.

As we have explained earlier, any configuration of the PCA $\mathcal{A}_p$ can be translated into a valid configuration of the resulting CA $\mathcal{A}_r$. Conversely, every valid configuration of $\mathcal{A}_r$ trivially corresponds to a configuration of the original one (applying the obvious reverse transformation).

Moreover, the rules of the automaton have been designed in such a way that in a valid configuration every cell could find its block, its orientation and therefore its facing and brother blocks, that are the ones that correspond to the facing and brother partitions in the partitioned automaton.

The only times when a cell cannot find its block in a valid configuration is when it is part of a blank block corresponding to a blank partition surrounded by blank partitions in the original PCA. In this case, the cell will do nothing (because it cannot determine whether or not the configuration is valid) which incidentally happens to be what it is supposed to do because the blank state is quiescent (so a blank partition surrounded by other blank partitions should remain blank).

After the blocks have been found, the switch and mix steps will occur correctly. The mix is the step that eventually applies the local rule of $\mathcal{A}_p$ (the switch has no incidence in a valid configuration, it is only used to guarantee conservation in an invalid configuration).

The simulation property can be expressed by using the simple translation function τ from the configurations of $\mathcal{A}_p$ into valid configurations of $\mathcal{A}_r$. For any configuration $\mathfrak{C}$ of $\mathcal{A}_p$, we have

$$F_p(\mathfrak{C}) = \tau^{-1}(F_r(\tau(\mathfrak{C})))$$

where F_p is the global transition rule of $\mathcal{A}_p$ and F_r that of $\mathcal{A}_r$.

6. Comments

6.1. Another Solution

There is a simpler solution if one wants only to convert a number-conserving and/or reversible PCA into a regular CA that conserves these properties.

The solution is to increase the number of states fourfold by adding a *directional layer*: if $\mathcal{Q}$ is the set of partitioned states of the original PCA, we make a CA working on the states $\mathcal{Q} \times \{\uparrow, \leftarrow, \downarrow, \rightarrow\}$.

A square cell of $\mathcal{A}_p$ is here transformed into a 2×2 square of cells, each holding a partitioned state and the arrow indicating the partition it corresponds to. The arrows do not change over time.

The behavior of the automaton is then similar to the one explained in the article but the needed neighborhood is smaller (a radius 2 neighborhood is sufficient because the configurations are more compactly represented and there are less possibilities for errors that have to be checked).

Fhe four directions are given the weights 0, $|\mathcal{Q}|$, $2|Q|$ and $3|\mathcal{Q}|$ respectively, and the weight of a cell is the sum of its state and the weight of its direction.

This method is simpler to implement but has many disadvantages over the one described in the article, most notably that it requires finite configurations to be translated into non finite ones (but still ultimately periodic) and that it increases the number of states.

6.2. Rotation Invariance

In the whole construction that we have described in this article, all four directions (up, left, down and right) have been considered similarly. Even the translation τ from configurations of $\mathcal{A}_p$ into configurations of $\mathcal{A}_r$ is rotation-invariant.

This means that if the PCA $\mathcal{A}_p$ has a rotation-invariant local rule then our resulting automaton $\mathcal{A}_r$ will also be. This is a good thing when devising simple PCA based on physical principles (both number-conservation and reversibility are often sought after because of physical considerations).

6.3. Other Partitions

In this article we have only considered two-dimensional 4-partitioned cellular automata. However there exist other kinds of partitioned cellular automata.

Important variants include for example one dimensional 2-partitioned and 3-partitioned cellular automata and two-dimensional 5-partitioned cellular automata.

For all of these different forms of PCA it is possible to adapt the construction explained in this article to obtained similar results. The key lies in the correct choice of the translation function τ that will define valid configurations.

Some important points are to be observed when choosing such a function:

- All resulting blocks must comprise the same number of cells. If not, the switch and mix steps are not conservative anymore.
- Marking blank states cannot be omitted. These states are more than mere fillers since they are the only way to determine the orientation of blocks. Markers must also be blank, not only to ensure that finite configurations have a finite total weight but also because the marking state must be quiescent (in case a cell is surrounded by cells in the marking state). In a number-conserving PCA all states are quiescent, but that is not necessarily the case for a reversible PCA.
- The translation pattern must be rotation invariant if rotation invariance is to be conserved by the transformation.
- Blocks must not necessarily be connex but non-connexity will probably increase the size of the required neighborhood in order to perform the necessary checks.

There are some simple translation functions that fit the aforementioned variants of PCA which make the results valid on these too. A smaller translation pattern might also exist for the 4-partitioned cellular automata considered in this article.

7. Conclusion

We have therefore shown how any partitioned cellular automaton can be converted into a regular cellular automaton in such a way that important properties such as number-conservation, reversibility and rotation invariance are conserved in the process.

The number of states is not increased (it is even decreased if we consider that a PCA has $\mathcal{Q}^4$ states where $\mathcal{Q}$ is the set of its partitioned states) but the considered neighborhood is.

The result is mainly interesting as a theoretical equivalence (there is no other real need to translate a PCA into a regular CA) because some constructions are much easier to perform on PCA. For instance, K. Morita was able to devise a PCA with only 3 states that is reversible, number-conserving, rotation invariant and Turing-complete. Our construction gives a regular CA that has all these same properties (although it works on a quite large neighborhood).

References

[1] U. Frisch, B. Hasslacher, and Y. Pomeau: Lattice-Gas Automata for the Navier-Stokes Equation, *Physica* **49D** (1991) 295-322.

[2] N. Margolus: Physics-Like Models of Computation, *Physica* **10D** (1984) 81-85.

[3] K. Morita and K. Imai: Number-Conserving Reversible Cellular Automata and their Computation-Universality, *Proceedings of MFCS'98 Workshop on Cellular Automata*, Brno (1998) 51-68.

[4] K. Morita, Y. Tojima and K. Imai: A Simple Computer Embedded in a Reversible and Number-Conserving Two-Dimensional Cellular Space, *Multiple-Valued Logic*, vol.6 (2001) 483-514.

[5] K. Morita, Y. Tojima, K. Imai and T. Ogiro: Universal Computing in Reversible and Number-Conserving Two-Dimensional Cellular Spaces, *Collision Based Computing*, Springer-Verlag (2002).

[6] K. Nagel and M. Schreckenberg: A Cellular Automaton Model for Freeway Traffic, *Journal of Physics* I, 2 (1992) 2221-2229.

Journées Automates Cellulaires 2008 (Uzès), pp. 141-160

RULE 110: UNIVERSALITY AND CATENATIONS

GAÉTAN RICHARD

Laboratoire d'informatique fondamentale de Marseille (LIF),
Aix-Marseille Université, CNRS;
39 rue Joliot-Curie, 13 453 Marseille, France
E-mail address: gaetan.richard@lif.univ-mrs.fr

Abstract. Cellular automata are a simple model of parallel computation. Many people wonder about the computing power of such a model. Following an idea of S. Wolfram [16], M. Cook [3] has proved than even one of the simplest cellular automata can embed any Turing computation. In this paper, we give a new high-level version of this proof using particles and collisions as introduced in [10].

Introduced in the 40s by J. Von Neumann as a parallel model of computation [13], *cellular automata* consist of many simple entities (*cells*) disposed on a regular grid. All cells evolve synchronously by changing their *state* according to the ones of their *neighbours.* Despite being completely known at the local level, global behavior of a cellular automaton is often impossible to predict (see J. Kari [6]). This comes from the fact that even "simple" cellular automata can exhibit a wide range of complex behaviors. Among those behaviors, one often refers as *emergence* the fact that "complexity" of the whole system seems far greater than complexity of its elements.

Elementary cellular automata are an example of subclass of "simple" cellular automata. They are obtained by considering only a one dimensional grid (i.e., a line of cells), two possible states and nearest neighbours (i.e., left and right one in addition to the cell itself). Although very restrictive, some elements of this class do exhibit very complex behaviors including emergence. One way to assert such a claim is to prove that some of those cellular automata can embed any Turing computation. Among elementary cellular automata, more likely candidate to this property were though to be the ones that exhibit meta-structures with predictable behavior. Those meta-structures have been studied with regards to their combinatorial aspect (see N. Boccara *et al.* [1] or J. P. Crutchfield *et al.* [5]) and widely used as support for constructions. In fact, M. Cook [3] managed to embed any Turing computation in an elementary cellular automaton (namely rule 110) using these structures. However, lack of formalism to manipulate those meta-structures forced him to develop long and complex combinatorial arguments to prove that intuition on behavior is correct.

In this paper, we shall use a new formalism on these meta-structures developed in [10] to provide a complete and high-level proof of Turing universality of rule 110 without the need of complex combinatorial arguments.

2000 ACM Subject Classification: 68Q80,68Q05,37F99.
Key words and phrases: Cellular automata, particles and collisions, Turing, simulation.

In section 1, we give formal definitions of cellular automata, discuss about the notion of Turing simulation and introduce the framework of particles and collisions. In section 2, we introduce the cyclic Post tag system (CPTS) used as an intermediate and prove this system is able of any Turing computation. Finally, In section 3, we explicitly give the meta-structures used, present in details the construction to encode CPTS and prove that the encoding method is valid.

1. Cellular automata

Cellular automata are a parallel computation model on a regular grid in discrete time. In general, this model is known to be able of any Turing computation. In this paper, we consider only a very simple subclass of cellular automata: elementary cellular automata (ECA). These automata are made of a line of cells with a binary state $\{0,1\}$ and only take into account the three nearest neighbours (i.e., left, center and right). An *elementary cellular automaton* is thus a function $f : \{0,1\}^3 \to \{0,1\}$ also called *local transition function.* One can notice that this class has only a finite number (256) of elements. Usually, those elements are referred by their index which is the integer obtained by taking for the i-th digit $f(i_0, i_1, i_2)$ where $i_0 i_1 i_2$ is the writing of i in base 2. In the rest of the paper, we focus on *rule 110* whose transition function is given on Table 1.

$f(l,c,r)$	0	1	1	0	1	1	1	0
(l,c,r)	$(1,1,1)$	$(1,1,0)$	$(1,0,1)$	$(1,0,0)$	$(0,1,1)$	$(0,1,0)$	$(0,0,1)$	$(0,0,0)$

Table 1: Local transition function of rule 110

ECA act, in a synchronous way, over *configurations* $c \in \{0,1\}^{\mathbb{Z}}$ by the *global transition function* $F(c)_i = f(c_{i-1}, c_i, c_{i+1})$. Starting from an *initial configuration* c_0, the sequence of successors $O = (F^{(i)}(c_0))_{i\in\mathbb{N}}$ is called the *orbit* starting from c_0. To draw an orbit of a cellular automaton, a convenient method it to pill up elements of this orbit leading to a *space-time diagram* (see Figure 1).

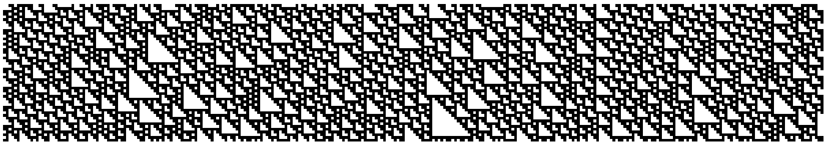


Figure 1: Example of a space-time diagram of rule 110 (time goes from bottom to top)

Due to the parallel nature of cellular automata, they have several differences with other computation models. Therefore, the notion of simulation is less straightforward than usual. Thus, it needs some discussions presented in the following.

1.1. Universalities

Among the differences between cellular automata and other systems, main ones are that they have no halting condition and act on infinite configurations. These two points make it very difficult to achieve a "natural" or even standard definition of simulation by cellular automata of other systems (and in particular Turing machines). In this section, we discuss the notion of Turing universality and present the version used in this paper.

To make a parallel between Turing machines and cellular automata, we first need to introduce some "halting" condition somewhere in cellular automata. This is generally achieved by seeking a particular word in the configuration. This seek can be done either at a specific place or anywhere on the configuration. Here, we choose the latter option leading to the following formal definition: A word $w \in \{0,1\}^*$ *occurs* in an orbit O if there exists $t \in \mathbb{N}$ and $x \in \mathbb{Z}$ such that $w = O_{t,x}O_{t,x+1}\dots O_{t,x+|w|-1}$.

For input, one solution is to restrict to finite configurations (i.e., configurations with a finite number of non 0 letters). However, this definition is too restrictive in our case. Therefore we prefer to use the less restrictive set of *ultimately periodic* configurations: that is configurations which are of the form ${}^{\omega}lmr^{\omega}$ where l, m and r are finite words. We request that these words can be easily computed from the input word of the Turing machine and that the resulting configuration "halts" if and only if the Turing machine halts. All those points can be formalized in the following definition:

Definition 1.1. A cellular automaton is *Turing universal* is for any Turing machine $\mathcal{M}$, there exists a word w and a log-space function f which maps any words $s \in \Sigma^*$ to three words l_s, m_s, r_s such that w occurs in the orbit starting from configuration ${}^{\omega}l_s m_s r_s^{\omega}$ if and only if $\mathcal{M}$ eventually halts on s.

One can note that our definition is a specific case of the more general scheme presented by B. Durand and Zs. Róka in [4].

Since most of the problems encountered with definitions of Turing universality come from the fact that we deal with two heterogeneous systems, another sensible approach of simulation is to focus on simulation of cellular automata by cellular automata, as described by N. Ollinger in [8]. This idea led to the notion of *intrinsic universality*. Intuitively, a cellular automaton simulates another one if the space-time diagram of the simulating one can be regularly embedded into the simulating one. Due to the fact that intrinsic universality requires the whole computation to be embedded in a regular way, intrinsic universality implies Turing universality but the converse is false. A more detailed study of those two types of universalities can be found in the survey made by N. Ollinger [9]. In this paper, we deal with the Turing universality of rule 110, the question of intrinsic universality of such a rule is still open.

Theorem 1.2 (M. Cook [3]). *Rule 110 is Turing-universal.*

The construction made by M. Cook makes heavy use of regular structures present in rule 110. However, due to a lack of specific formalism, it fail to achieve the proof on these structures level and must default to a technical combinatorial approach. In this paper, we intend to lever the proof by using specific high-level tools on these meta-structures. These meta-structures and tools are presented in the next part.

1.2. Particles and collisions

Through very restricted, elementary cellular automata can exhibit a wide range a behaviors. Those behaviors have been experimentally categorised by S. Wolfram [15] into four classes (see Figure 2): Class I regroups cellular automata whose behavior converges towards a stable configuration. Class II is constituted by those whose orbits ultimately go into a cycle. Class III regroups the ones whose behavior seems random and does not exhibit any kind of regularity. At last, elements of class IV are cellular automata where *"(...) localized structures are produced which on their own are fairly simple, but these structures move around and interact with each other in very complicated ways. (...)"*. Such phenomenon is often referred as *self-organisation* and is though to include a great computational power. In fact, simulation of Turing machine by rule 110 heavily relies on such structures. To use those structures, we need a formalism as the one introduced in [10] which gives us a formal support on intuitive tools.

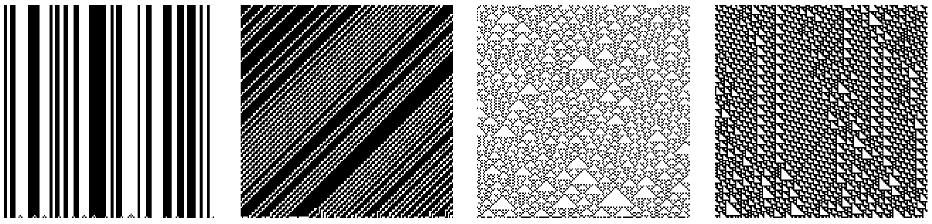

Class I (rule 232) Class II (rule 215) Class III (rule 18) Class IV (rule 110)

Figure 2: Behaviors of elementary cellular automata

Intuitively, those elements (which can be seen in the last class on Figure 2) can be easily described: most of the space-time diagram is filled with a bi-periodic pattern called background (Figure 3a). Among backgrounds, some uni-periodic structures called particles seem to travel (Figure 3b). These particles interact with each other and give birth to new particles in collisions (Figure 3c).

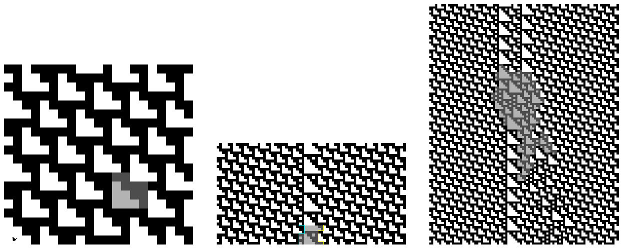

(a) Background (b) Particle (c) Collision

Figure 3: Examples of elements present in self-organisation and symbolic representation.

To give a formalism of these object, one heavily relies on two-dimensional aspect of space-time diagram. Therefore, in the rest of the paper, we only consider bi-infinite space-time diagrams (i.e., orbits with an infinite sequence of predecessors). Moreover, all definitions are based on discrete two-dimensional geometry. In this vision, space-time diagrams are elements of $\{0,1\}^{\mathbb{Z}^2}$ with constraints induced by local transition function.

A *coloring* is an application $\mathcal{C}$ from a subset $\mathrm{Sup}(\mathcal{C})$ of $\mathbb{Z}^2$ to $\{0,1\}$. If $\mathrm{Sup}(\mathcal{C})$ is finite then the coloring is said to be *finite*. Restriction of a coloring $\mathcal{C}$ to a subset S of $\mathbb{Z}^2$ is denoted as $\mathcal{C}_{|S}$. Translation of a coloring along a vector u is the coloring of support $\{s+u|s \in \sup(\mathcal{C})\}$, defined by $(u \cdot \mathcal{C})(z+u) = \mathcal{C}(z)$. Disjoint union of two colorings $\mathcal{C}$ and $\mathcal{C}'$ whit $\mathrm{Sup}(\mathcal{C}) \cap \mathrm{Sup}(\mathcal{C}') = \emptyset$ is defined such that $z \in \mathrm{Sup}(\mathcal{C})$, it holds $\mathcal{C} \oplus \mathcal{C}'(z) = \mathcal{C}(z)$ and for all $z \in \mathrm{Sup}(\mathcal{C}')$, it holds $\mathcal{C} \oplus \mathcal{C}'(z) = \mathcal{C}'(z)$.

A *background* is a triplet $\mathfrak{B} = (\mathcal{C}, u, v)$ where u, v are two non-collinear elements of $\mathbb{Z}^2$ and $\mathcal{C}$ a finite coloration satisfying that $\bigoplus_{i,j \in \mathbb{Z}^2}(iu+jv) \cdot \mathcal{C}$ is a space-time diagram. In the rest of the paper, we abusively also denote by $\mathfrak{B}$ the resulting space-time diagram. A *particle* is a tuple $\mathfrak{P} = (\mathcal{C}, u, \mathfrak{B}_l, \mathfrak{B}_r)$ where $\mathcal{C}$ is a finite coloring, $u \in \mathbb{Z}^2$, $\mathfrak{B}_l$ and $\mathfrak{B}_r$ are backgrounds, provided that $\mathcal{I} = \bigoplus_{k \in \mathbb{Z}} ku \cdot \mathcal{C}$ separate the plan in two 4-connected zone L and R (oriented according to u) ensuring that $\mathfrak{B}_{|L} \oplus \mathcal{I} \oplus \mathfrak{B}'_{|R}$ is a space-time diagram[1]. At last, a *collision* is a pair $(\mathcal{C}, L)$ where $\mathcal{C}$ is a finite coloring, L is a finite sequence of n particles $\mathfrak{P}_i = (\mathfrak{B}_i, \mathcal{C}_i, u_i, \mathfrak{B}'_i)$, satisfying:

(1) $\forall i \in \mathbb{Z}_n, \quad \mathfrak{B}'_i = \mathfrak{B}_{i+1}$;
(2) $\mathcal{I} = \mathcal{C} \oplus \bigoplus_{i \in \mathbb{Z}_n, k \in \mathbb{N}} ku_i \cdot \mathcal{C}_i$ cut the plan in n 4-connected zones;
(3) For all $i \in \mathbb{Z}_n$, $\mathcal{C} \oplus \bigoplus_{k \in \mathbb{N}} (ku_i \cdot \mathcal{C}_i \oplus ku_{i+1} \cdot \mathcal{C}_{i+1})$ cut the plane in two 4-connected zones. Let P_i be the one right of $\mathfrak{P}_i$;
(4) $\mathfrak{C} = \mathcal{I} \oplus \bigoplus_i \mathfrak{B}_{i|P_i}$ is a space-time diagram.

Since finite colorings involved in particles and in collisions can be quite large, it would be unreadable to give them in an analytic form. That's why we depict them using a graphical version. To help the reader convince itself, rather than just depicting the coloring, we give the finite coloring "in context" and highlight it. This representation is more intuitive but nevertheless completely defines the object. In this paper, all background, particles and collisions shall be given this way (see figure 3). For collisions, we also give the name of involved particles.

The idea behind formalism [10] is to manipulate space-time diagram representing particles as lines and collisions as points. Such representation allows to represent evolutions of the cellular automaton as a planar map (see for example Figure 15) and is formalized below:

Definition 1.3. a *catenation scheme* is a planar map whose vertices are labeled by collisions and edges by particles which are coherent with respect to collisions.

A catenation scheme is a high-level symbolic assembly of particles and collisions. To make this scheme correspond to a valid space-time diagram, one needs to give explicit positions for every vertex and check that all local constrains are correct. Alternatively, those positions can be given indicating relative position of collisions, for example by specifying the number of repetitions of particles. Such set of repetitions is called *valid affectation* is the resulting object is a space-time diagram. The main point is that, from a catenation scheme, one can automatically know the form of the set of valid affectations.

Theorem 1.4 ([10]). *Given a finite catenation scheme, the set of valid affectations is a computable semi-linear set.*

[1]In an exact version, disjoint union is replaced by patchwork which require the different colorings to have a "safety border" on which they agree. This condition can be easily fulfilled by making the finite coloring larger. In this paper, we stick to this simplified version.

Catenations allow us to construct complex behaviors for cellular automata exhibiting self-organisation. To apply this in the case of rule 110, we need to manually extract a set of particles and collisions and then use it to simulate our Turing machine. Although rule 110 is known to have a very wide and complex system of particles and collisions, we are still not able to simulate directly a Turing machine. To make the simulation, M. Cook introduces an intermediate dynamical system known as cyclic Post tag systems with several additional constraints. To do this, one must first prove that this system can embed any Turing computation and then that rule 110 can embed this system.

For completeness of the proof, the next section describe how Turing machines can be embedded into this specific version of cyclic Post tag system. On first reading, the reader who is mainly concerned with rule 110 stuff can easily skip this part and just read cyclic Post tag systems definition (def 2.2) and assume the result of proposition 2.4.

2. Cyclic Post tag systems

In this section, we prove that cyclic Post tag systems with additional restrictions can simulate any Turing machine. Those systems are a variant of Post tag systems whose Turing power has been know for long (see H. Wang [14] or J. Cocke and M. Minsky [2]). From those systems, obtaining a cyclic one is not very difficult but ensuring the additional restrictions require a deep understanding of the simulation. Since those restrictions are a key point of the proof of rule 110, we choose to give a full proof of Turing machines simulation.

2.1. Definitions

Introduced by A. Turing in the 30s [12], *Turing machines* are one of the main dynamical models of computation. In our case, they consist of a bi-infinite *tape* filled with letters chosen among a finite alphabet Σ (see Figure 4). On the tape, there is a unique *head* with a *state* taken among a finite set Q. Dynamics is obtained the following way: at each time step, the head can write a new letter at its position, change its state and make a move to the left or to the right. The behavior of the head is uniquely determined by the current state of the head and the symbol on the tape under it. This behavior is denoted by the *transition function* δ. Initially, the tape filled with one distinguished *white* letter $w(\in \Sigma)$ on all but a finite portion called *input*. The head starts at position 0 in the *initial state* q_0 . Computation steps are done by applying the local rule until the head enters the *halting state* q_f. This can be formalized with the following definition:

Definition 2.1. A *Turing machine* (TM for short) is a tuple $(\Sigma, w, Q, q_0, q_f, \delta)$ where:

- Σ is the finite set of *letters*;
- $w \in \Sigma$ is the *white* letter;
- Q is the finite set of *states*;
- $q_0, q_f \in Q$ are the *initial* (resp. *halting*) state;
- $\delta : Q \times \Sigma \to Q \times \Sigma \times \{\leftarrow, \rightarrow\}$ is the *transition function*.

At each step, the system is fully defined by a *configuration* consisting of the non-white portion of the tape along with the current state and position of the head. In this system, all changes are localised under the head. Although simple, this system has as much computational power as most of other known dynamical systems. In this paper, we also use another system known as *cyclic Post tag system*. This system was first introduced by

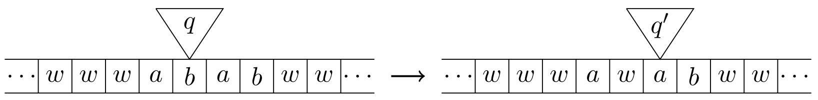


Figure 4: Example of Turing machine transition $\delta(q, b) = (q', w, \rightarrow)$

E. Post in 1943 [11]. A *Post Tag system* (PTS for short) can be described as a finite queue on a finite alphabet Σ. At each step, the system pops a finite fixed number n of letters from the queue. Then, according to the first popped letter, it pushes a (possibly empty) word at the end of the queue (see Figure 5a). The function associating the pushed word to the read letter δ is called *transition function.* The system starts with an initial finite *input* word in the queue. Transition rule is applied until there are not enough letters left to pop in the queue (i.e., strictly less than n). Formally, a *Post Tag system* can be depicted as a triplet (Σ, n, δ) where Σ is the finite set of *letters*; n is a non-null integer and $\delta : \Sigma \rightarrow \Sigma^*$ is the *local transition function.*

01 01100
01 100
10 0
0*1 00*
00
ϵ (halts)

(a) Post Tag System $(\{0,1\}, 2, \theta)$ with $\theta(0) = \epsilon$ and $\theta(1) = 100$

11011	$(\epsilon, 10011, 011100, 01)$
1011	$(10011, 011100, 01, \epsilon)$
011*10011*	$(011100, 01, \epsilon, 10011)$
11110011	$(01, \epsilon, 10011, 011100)$
1110011*01*	$(\epsilon, 10011, 011100, 01)$
	...

(b) Cyclic Post Tag System $(\epsilon, 10011, 011100, 01)$

Figure 5: Example of Post tag systems transitions

In this paper, we use a variant of this system, introduced by M. Cook [3], called *Cyclic Post Tag System* (CPTS for short) depicted in Figure 5b. In this variant, the alphabet is fixed to $\{0, 1\}$ and the transition rule is replaced by a finite cyclic list of words $(w_0, w_1, \ldots, w_{k-1})$ on $\{0,1\}^*$. At each step, the systems pops the first letter. If this letter is 1, it pushes the first word of the list (here w_0) at the end of the queue. Then, in all cases, it rotates the list of words — here, for example, the list becomes $(w_1, w_2, \ldots, w_{k-1}, w_0)$. As previously, starting from an initial *input* word in the queue, transitions occur until the queue is empty. In this case, the system is said to *halt* on the selected input. This leads to the following definition:

Definition 2.2. A *cyclic Post tag system* $\mathcal{P}$ is a finite cyclic list $(w_0, w_1, \ldots, w_{k-1})$ of words over the alphabet $\{0, 1\}$.

Once again, at each step, the system can be entirely characterised by a *configuration* consisting of the current content of the queue and the current rotation of the cyclic list (more precisely, the index of the first word in the list). The rest of this section is devoted to prove that CPTS can embed any Turing machine even when ensuring two additional restrictions. The first one is on the length of every word in the cyclic list. The other one is on the occurrence of letter 1 during any Turing simulation.

2.2. From Turing machines to cyclic Post tag systems

In this section, we show how a CPTS can simulate a Turing machine. Intuitively, the notion of simulation indicates that there exists an easy way to transform any input of the Turing machine into an input of the CPTS such that the Turing machine halts on the input if and only if the CPTS does. This can be formalized by the following:

Definition 2.3. A CPTS $\mathcal{P}$ simulates a Turing machine $\mathcal{M}$ if there exists a function $f : \Sigma^* \to \{0,1\}^*$ which is simple[2] such that for any word $s \in \Sigma^*$, $\mathcal{M}$ eventually halts on s if and only if $\mathcal{P}$ eventually halts on $f(s)$.

With this definition, we can state the main theorem of this section which says that a subset of CPTS is sufficient to simulate any Turing machine. As underlined before, the result has been already known for long in the general case but restrictions need a deep understanding of the method used. For this reason and to give the reader a complete view of the embedding process, we give in the following the complete reduction. Restrictions may seem quite mystic by now but they will appear when embedding this system into cellular automaton 110.

Proposition 2.4. *Any Turing machine $\mathcal{M}$ can be simulated by a cyclic Post tag system $\mathcal{P}$ such that:*

- *the length of any word in $\mathcal{P}$ is a multiple of* 6*;*
- *during any step of the simulation, there is at most K consecutive steps with* 0 *as the first popped letter. Moreover, K only depends on $\mathcal{M}$.*

Proof. First of all, it is well known than any Turing machine can be simulated by a Turing machine on alphabet $\{0,1\}$ with white letter 0. Since the reduction is trivial, we restrict ourselves to Turing machine with this alphabet. In this proof, we prove an even stronger result: there exists a transformation of any Turing machine configurations into CPTS configurations which commutes with dynamics. The proof is done by using PTS as an intermediate model.

Let us take any Turing machine $\mathcal{M} = (\{0,1\}, 0, Q, q_0, q_f, \delta)$ and c be a configuration of the Turing machine. It can be entirely defined by the state of the head q , the portion of the tape on the left of the head, the one on the right and the letter under the head i (see Figure 6). Since left and right words have only a finite number of 1 letters (i.e., non-white), they can be depicted as integers n_l and n_r which give c on the form (n_l, n_r, q, i).

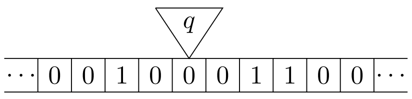
At this point, let us how encode this configuration into a PTS (Σ, n, δ') configuration. To encode all information, we use different methods:

- the left (resp. right) integer is encoded in unary between start and end markers;
- state is encoded in every letter of the queue;
- current letter is encoded using the fact that only one letter over n is read.

To allow encoding of all these information, let us take an alphabet Σ_0 on the form $M \times \{◂, ▸\} \times Q \times \{0,1\}$ where $M = \{▷, •, ⧏, ◀\}$ contains a marker information (respectively one for start, one for interior and two for end). The part $\{◂, ▸\}$ indicates if we speak about left or right word. the set Q is used to encode the current state. The last part of the alphabet refers to the parity of the position: throughout the construction, every letter on the form $(m, f, q, 0)$ is followed by a $(m, f, q, 1)$ letter. Such sequence is called *representation* and depicted as m_f (omitting the state for clarity). Moreover, representations are often

[2] usually, we request a log-space function but here we use the even more restrictive notion of morphism

doubled, the result is called *tag*. With this formalism, a configuration $c = (n_l, n_r, q, 0)$ is encoded in the queue as $\triangleright_{\blacktriangleleft}\triangleright_{\blacktriangleleft}(\bullet_{\blacktriangleleft}\bullet_{\blacktriangleleft})^{2n_l}\mathring{\triangleleft}_{\blacktriangleleft}\mathring{\triangleleft}_{\blacktriangleleft}\dot{\triangleleft}_{\blacktriangleleft}\dot{\triangleleft}_{\blacktriangleleft}\quad \triangleright_{\blacktriangleright}\triangleright_{\blacktriangleright}(\bullet_{\blacktriangleright}\bullet_{\blacktriangleright})^{2n_r}\mathring{\triangleleft}_{\blacktriangleright}\mathring{\triangleleft}_{\blacktriangleright}\dot{\triangleleft}_{\blacktriangleright}\dot{\triangleleft}_{\blacktriangleright}$ (see also Figure 6). For the $(n_l, n_r, q, 1)$ case, the encoding is the same except that the first letter is removed. As we have made four letters blocs, we choose $n = 4$ (i.e., read one letter and then discard 3). This ensure that exactly one letter is read per tag.



Turing configuration	
With left and right words	$(10, 110, q, 0)$
Converting into integers	$(2, 6, q, 0)$
PTS configuration (state omitted)	$\triangleright_{\blacktriangleleft}\triangleright_{\blacktriangleleft}(\bullet_{\blacktriangleleft}\bullet_{\blacktriangleleft})^{4}\mathring{\triangleleft}_{\blacktriangleleft}\mathring{\triangleleft}_{\blacktriangleleft}\dot{\triangleleft}_{\blacktriangleleft}\dot{\triangleleft}_{\blacktriangleleft}\quad \triangleright_{\blacktriangleright}\triangleright_{\blacktriangleright}(\bullet_{\blacktriangleright}\bullet_{\blacktriangleright})^{12}\mathring{\triangleleft}_{\blacktriangleright}\mathring{\triangleleft}_{\blacktriangleright}\dot{\triangleleft}_{\blacktriangleright}\dot{\triangleleft}_{\blacktriangleright}$

Figure 6: From Turing machine configuration to PTS configuration

At this point, let us study how transitions are achieved. Let (n_l, n_r, q, i) be a configuration of the Turing machine. Let us assume that the transition is on the form $\delta(q, i) = (q', s, \leftarrow)$ (the case $\rightarrow$ is obtained by symmetry). In these conditions, the successor of the configuration is $(n_l/2, 2n_r + s, q', n_l \mod 2)$ (shifting bits corresponds to multiplying or dividing by 2). Thus, in our PTS we need to:

- update the state;
- multiply the right value by 2;
- add s to the right value;
- divide the left value by 2;
- make an integer floor on the left value;
- read the modulus of the left value and transform it into positioning.

Sadly, doing all these operations requires three passes. The current pass is encoded in all letters of the queue by extending the alphabet with a Cartesian product: the new alphabet is $\Sigma_1 = \Sigma_0 \times \{A, B, C\}$. In pass A, we do the first four points. In pass B, we do the floor and read the modulus and in pass C we convert the read modulus to alignment.

Now, let us construct the transition function achieving this behavior. A full example of transition is given in Figure 7. During pass A, for each encountered tag, we know the current state (present in the letter) and the current value under the head (read in the last element of the letter). Thus we know the transition and can update the state and the left and right values. For the doubled value, we copy the start and end tags, double each interior tag and add one interior case if 1 if written. For the divided value and for each encountered tag (start, interior or end), we write only one corresponding representation (thus dividing the number of letters by two even for boundaries). As the word is only made of tags, the relative position of read letters is the same after the first pass. During pass B, floor and modulus of the divided word are computed as depicted in Figure 7: First, the starting letter is read whereas eating one interior representation and writing a start tag. Thereafter, we write one interior tag for every two interior representations. At the end, we can either arrive on the first or second ending representation according to the interior representation parity. In addition to write both end tags, we can also insert some new letters # to change the relative position in the next step. On other portions, we just need to make a copy which can be easily achieved since they are compound of tags (recall tags are made of four letters and thus are read exactly one time). The last part also consists just of copying. Passing above letters # ensures the correct new positioning of read letters. A careful reader may note that the depicted method is not fully correct since in pass B, we must know which word must be floored and have no longer access to this information. However, this can be easily overcome by enlarging once again the alphabet.

The last point is to define the behavior in the case we reach an halting state of the Turing machine. This case is easily dealt with by requesting the Post system to erase the whole queue, causing it to halt. To ensure our additional condition, we need to bound the number of step without writing. Since the only case where there is no writing is when erasing the configuration on halting, it is easy to obtain a bound by requesting that the simulated Turing machine halt only with empty tape. This set of machine is obviously also Turing powerful. With this restriction, the number of step without write is bounded by the number of steps (precisely six) to erase the empty configuration $\triangleright_{\blacktriangleleft}\triangleright_{\blacktriangleleft}\lhd_{\blacktriangleleft}\lhd_{\blacktriangleleft}\blacktriangleleft_{\blacktriangleleft}\blacktriangleleft_{\blacktriangleleft}\triangleright_{\blacktriangleright}\triangleright_{\blacktriangleright}\lhd_{\blacktriangleright}\lhd_{\blacktriangleright}\blacktriangleleft_{\blacktriangleright}\blacktriangleleft_{\blacktriangleright}$.

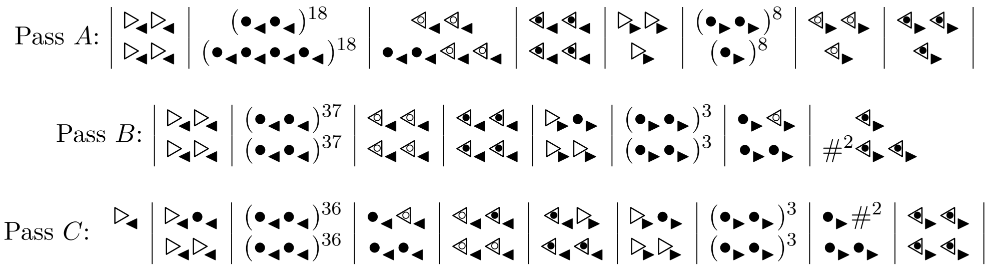


Figure 7: Example of a Turing transition simulation (above is the read queue and below the corresponding elements written).

At this point, we want to convert our PTS system into a CPTS one. This can be done without any real difficulty (see Figure 8). Let us take the alphabet Σ of the PTS. It is possible to represent the n-th letter with a fixed length sequence of 0 by marking a letter 1 in a the n-th position. One can trivially request that the length m of those words is a multiple of 6 (ensuring our first additional condition). This way, we can convert all letters to words on alphabet $\{0, 1\}$. Since all letters have the same size, each transition read a fixed number $4m$ of letters. The transition is indicated by the 1 in the first m read letters. Thus the cyclic list can be done the following way: take a list of length $4m$, for the first m words, the word is the result by the transition of the PTS on the m-th letter. All other words are taken empty. To end this, let us look at the maximal number of consecutive erasing. Since a complete rotation of the list correspond to a transition of our PTS, we know there are at most $24m$ steps without writing.

Initial Post system	$a \vdash bb, b \vdash caa, c \vdash \epsilon$
Representation of letters	$a \Leftrightarrow 100000, b \Leftrightarrow 010000, c \Leftrightarrow 001000$
Representation of images	$bb \Leftrightarrow 010000010000, caa \Leftrightarrow 001000100000100000, \epsilon \Leftrightarrow \epsilon$
Cyclic word list (with step 4)	$(010000010000, 001000100000100000, \epsilon, \underbrace{\epsilon, \ldots, \epsilon}_{18\times})$

Figure 8: From PTS to CPTS

■

In this transformation, we need to go thorough the configuration three times to simulate one transition of the initial Turing machine. Since we use unary encoding, the length of the configuration can double at each simulated step, thus the speed of simulation suffers an exponential slowdown. With a more subtle and complex methods, T. Neary and D. Woods managed to obtain a polynomial slowdown [7, 17] . This result allows them to obtain stronger results using the simulation presented in the rest of the paper. In particular, they prove that predicting rule 110 is P-complete with respect to Turing reducibility.

3. Universality of rule 110

Now, let us go back to rule 110. This last section is devoted to simulate a CPTS (with the additional restrictions) with rule 110. Of course, this simulation is done using tools introduced in section 1.2. With those tools, the way of constructing and proving the simulation is the following:

(1) First, give an explicit set of particles and collisions of rule 110;
(2) Then, construct the simulation at a global level using catenations;
(3) Last, use properties of catenation to ensure that simulation is correct (in particular at local level).

3.1. Particles and collisions of rule 110

Rule 110 do possess a very large number of particles and collisions. In this part, we extract a small subset of those particles and collisions that are used in the construction.

For background, we use only one background (the standard one on rule 110) which is given in Figure 9. Since it is the only background, it is omitted in all following objects and representations.

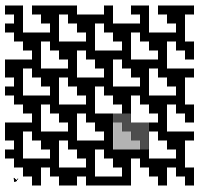


Figure 9: Background used in the construction (coloring is highlighted)

During the construction, we use a bunch of particles and collisions. To ease the reading and understanding of the construction, some hints about particles and collisions used are given alongside their description.

These particles serve as support for information. Dynamic is achieved by a set of 23 collisions depicted in Figures 11 to 13. As for particles, each collision is given by an extract of the space-time diagram where non perturbation pattern is highlighted. Moreover, symbolic behavior of collisions on particles are also given as formulae.

Remark 3.1. The full set of particles and collisions used in this paper contains 18 particles and 23 collisions that are all depicted in Figures 10, 11, 12 and 13. ■

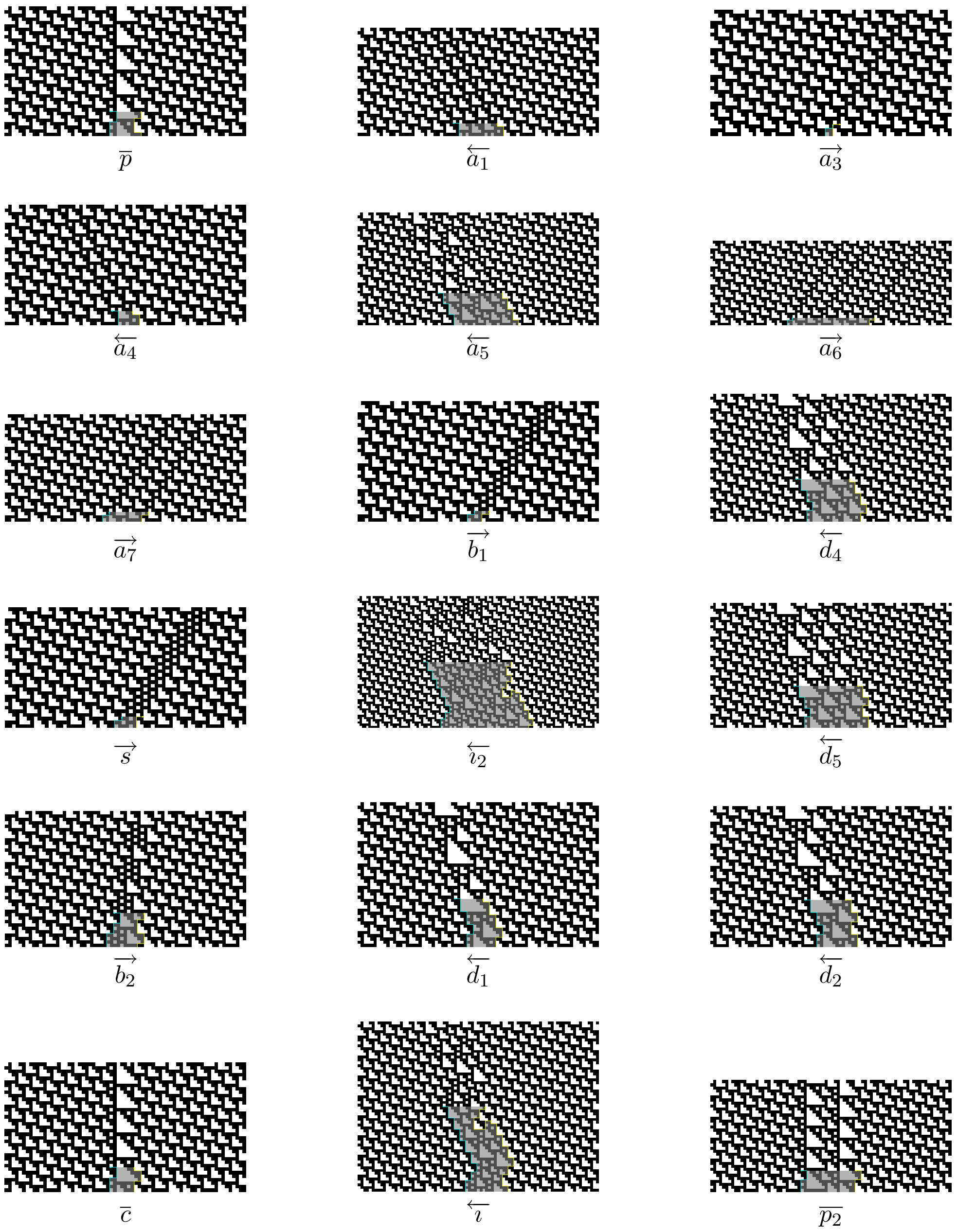
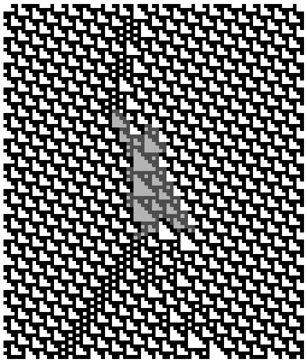
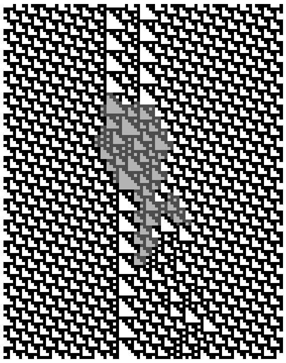
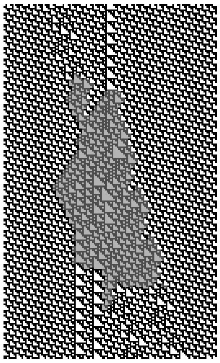
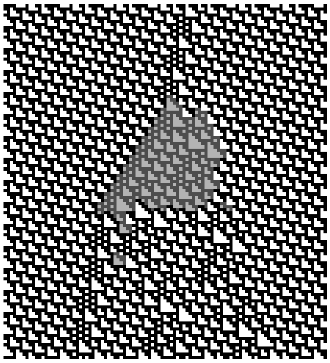


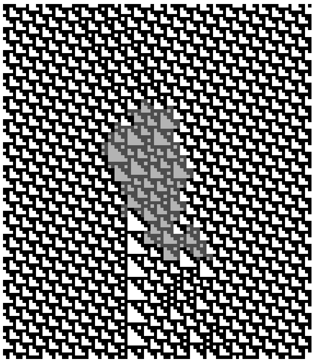
Figure 10: Particles used in the construction (with highlighted finite coloring)

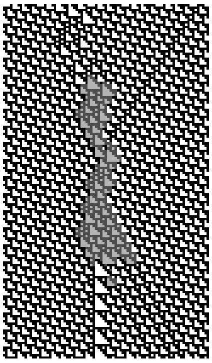

f
$\overrightarrow{b_1} + \overleftarrow{\imath} \vdash \overrightarrow{b_2}$

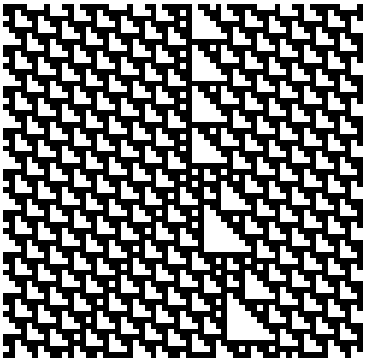

d
$\overline{p} + \overleftarrow{\imath} \vdash \overline{p_2}$

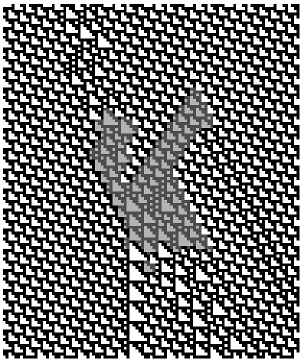

d'
$\overline{p_2} + \overleftarrow{\imath_2} \vdash \overleftarrow{\imath} + \overline{p}$


f'
$\overrightarrow{b_2} + \overleftarrow{\imath_2} \vdash \overrightarrow{b_2}$


g_1
$\overline{c} + \overleftarrow{\imath} \vdash \overleftarrow{a_1}$


g_2
$\overline{c} + \overleftarrow{a_1} \vdash \overleftarrow{\imath} + \overrightarrow{a_3}$


g_5
$\overrightarrow{a_3} + \overleftarrow{d_1} \vdash \overline{p}$


g_4
$\overline{c} + \overleftarrow{a_5} + \overleftarrow{a_4} \vdash \overleftarrow{\imath}$

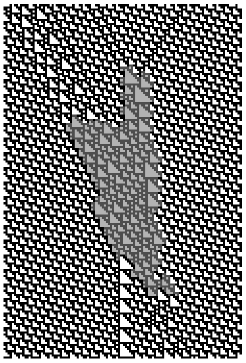
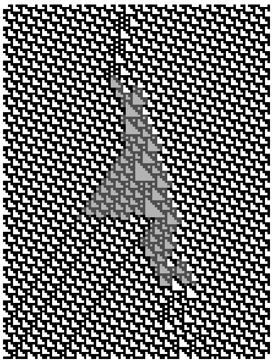
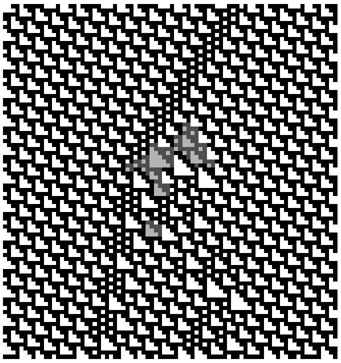

g_3
$\overline{c} + \overleftarrow{\imath} \vdash \overleftarrow{a_5} + \overleftarrow{a_4} + \overrightarrow{a_3}$

Figure 11: Collisions used in the construction



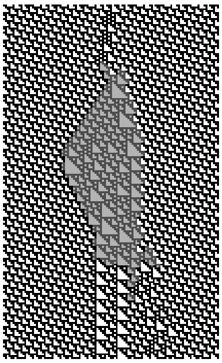
g'_7

$\overrightarrow{a_6} + \overleftarrow{\imath} \vdash \overrightarrow{b_2}$



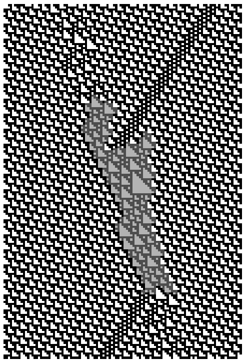
g'_8

$\overrightarrow{b_2} + \overleftarrow{\imath} \vdash \overrightarrow{b_1}$



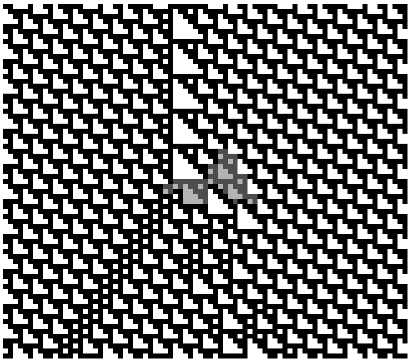
g_8

$\overline{p_2} + \overleftarrow{\imath} \vdash \overrightarrow{b_2}$



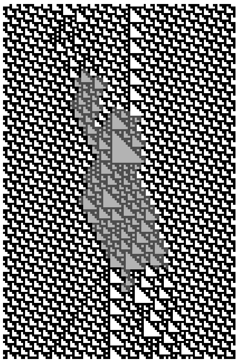
h

$\overrightarrow{s} + \overleftarrow{\imath} \vdash \overleftarrow{\imath} + \overrightarrow{s}$



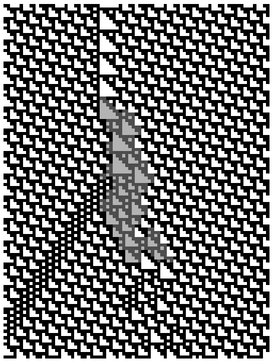
g'_9

$\overrightarrow{b_1} + \overleftarrow{\imath} \vdash \overline{p}$



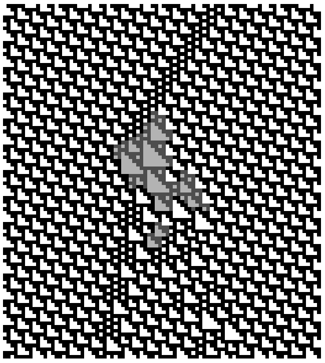
g_6

$\overrightarrow{a_3} + \overline{p} + \overleftarrow{d_4} \vdash \overleftarrow{\imath} + \overline{p}$



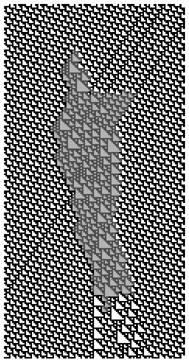
k

$\overrightarrow{s} + \overleftarrow{\imath} \vdash \overline{c}$



s

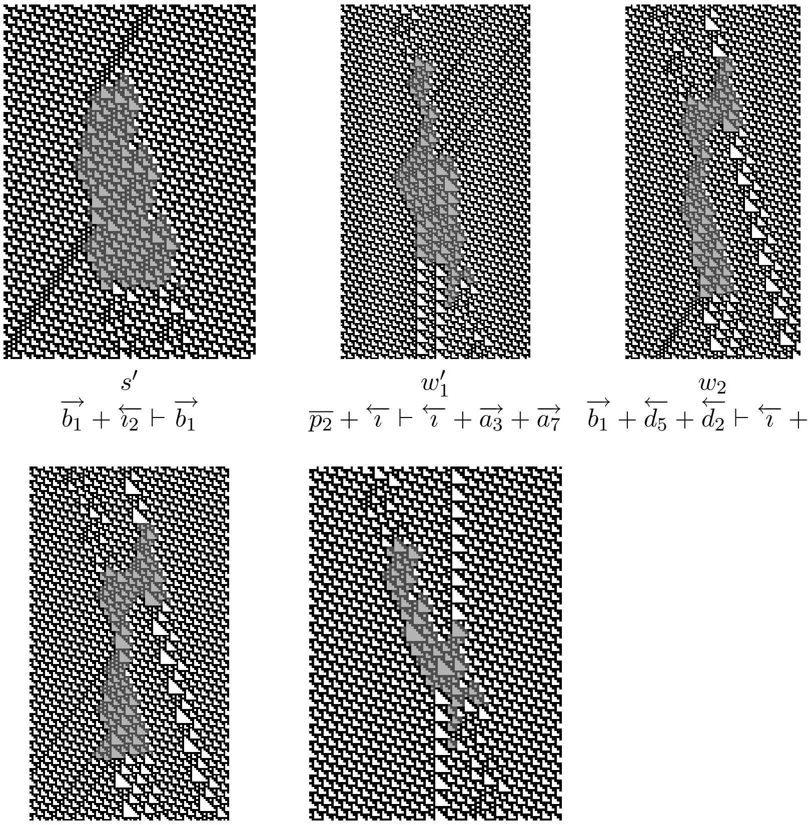
$\overrightarrow{b_2} + \overleftarrow{\imath} \vdash \overrightarrow{b_1}$



g'_6

$\overrightarrow{a_3} + \overline{p} + \overleftarrow{d_4} \vdash \overrightarrow{a_6}$

Figure 12: Collisions used in the construction (cont.)



s'
$\overrightarrow{b_1} + \overleftarrow{\imath_2} \vdash \overrightarrow{b_1}$

w_1'
$\overline{p_2} + \overleftarrow{\imath} \vdash \overleftarrow{\imath} + \overrightarrow{a_3} + \overrightarrow{a_7}$

w_2
$\overrightarrow{b_1} + \overleftarrow{d_5} + \overleftarrow{d_2} \vdash \overleftarrow{\imath} + \overleftarrow{d_1}$

w_2'
$\overrightarrow{a_3} + \overrightarrow{a_7} + \overleftarrow{d_5} + \overleftarrow{d_2} \vdash \overleftarrow{\imath} + \overleftarrow{d_1}$

j
$\overline{c} + \overleftarrow{\imath} \vdash \overleftarrow{\imath} + \overline{c}$

Figure 13: Collisions used in the construction (end)

At this point, let us discuss how to encode CPTS elements using particles. Encoding information into particles can be done in two ways: either by the type of particle used or by the relative position in a group of particles. In the construction, both methods are used. This implies, in particular, that some bits of information are conveyed by groups of parallel particles. Those groups of particles are called *symbols* and named with capital letters. To encode binary letters of the CPTS, we use groups of four particles, the letter $x \in \{0, 1\}$ is encoded by the relative position of those particles. In the construction different groups are used to encode letters: $\overleftarrow{F}_x = (\overleftarrow{\imath}\,\overleftarrow{\imath_2})^4$ are words list letters (called *frozen letters*); $\overline{C}_x = \overline{c}^4$ are *queue letters* and $\overleftarrow{W}_x = \overleftarrow{\imath}^4$ temporary container (called *unfrozen letters*). To encode the cyclic list of words, we also use a *starting* symbol $\overleftarrow{S} = \overleftarrow{\imath}\,\overleftarrow{d_1}\overleftarrow{d_4}\overleftarrow{\imath}^4\overleftarrow{\imath_2}$ and a *delimiter* $\overleftarrow{D} = \overleftarrow{d_5}\overleftarrow{d_2}\overleftarrow{d_1}\overleftarrow{d_4}\overleftarrow{\imath}^4\overleftarrow{\imath}$. The behavior of transition is stored in a *erasing* symbol $\overrightarrow{B} = \overrightarrow{b_2}$ or a *copying* symbol $\overline{P} = \overline{p}$. For the dynamic, two additional symbols are needed: a *clock* $\overrightarrow{T} = \overrightarrow{s}^4$ and some *junk* $\overleftarrow{J} = \overleftarrow{\imath}^2$. Some of these symbols also have a *degraded* version that we denote with a tilde: $\tilde{F}_x = (\overleftarrow{\imath}\,\overleftarrow{\imath_2})^3(\overleftarrow{\imath}\,\overleftarrow{\imath})$, $\tilde{S} = \overleftarrow{\imath}\,\overleftarrow{d_1}\overleftarrow{d_4}\overleftarrow{\imath}^4\overleftarrow{\imath}$, $\tilde{D} = \overleftarrow{d_5}\overleftarrow{d_2}\overleftarrow{d_1}\overleftarrow{d_4}\overleftarrow{\imath}^4\overleftarrow{\imath}$, $\tilde{P} = \overrightarrow{a_3}\overrightarrow{a_7}$ and $\tilde{B} = \overrightarrow{b_1}$.

Combining collisions into finite catenations, it is possible to obtain 10 different possible behaviors with symbols (and 6 additional with altered versions). The complete list of such catenations are given in Figure 14 and Figure 15. Altered versions are not fully depicted since they can be very easily obtained from non-altered ones. The catenation $\tilde{\mathfrak{C}}$ is the same as $\mathfrak{C}$ with half number of collisions. For $\tilde{\mathfrak{S}}_0$, the only difference is that the upper collision is g'_8 rather than f' (as for $\tilde{\mathfrak{M}}_0$). For $\tilde{\mathfrak{S}}_1$ and $\tilde{\mathfrak{M}}_1$, the same holds replacing the upper collision d' by w'_1. The last catenations $\mathfrak{E}_0$, $\tilde{\mathfrak{E}}_0$, $\mathfrak{E}_1$ and $\tilde{\mathfrak{E}}_1$ are made with only one collision (w_2 for the first two ones, w'_2 for the last two ones).

$\mathfrak{N}$	$\overrightarrow{T} + \overleftarrow{W}_x \vdash \overline{C}_x$		
$\mathfrak{C}'$	$\overrightarrow{T} + \overleftarrow{J} \vdash \overleftarrow{J} + \overrightarrow{T}$		
$\mathfrak{C}$	$\overline{C}_x + \overleftarrow{W}_y \vdash \overleftarrow{W}_y + \overline{C}_x$	$\tilde{\mathfrak{C}}$	$\overline{C}_x + \overleftarrow{J} \vdash \overleftarrow{J} + \overline{C}_x$
$\mathfrak{S}_0$	$\overline{C}_0 + \overleftarrow{S} \vdash \overleftarrow{J} + \overrightarrow{B}$	$\tilde{\mathfrak{S}}_0$	$\overline{C}_0 + \tilde{S} \vdash \overleftarrow{J} + \tilde{B}$
$\mathfrak{S}_1$	$\overline{C}_1 + \overleftarrow{S} \vdash \overleftarrow{J} + \overline{P}$	$\tilde{\mathfrak{S}}_1$	$\overline{C}_1 + \tilde{S} \vdash \overleftarrow{J} + \tilde{P}$
$\mathfrak{M}_0$	$\overrightarrow{B} + \overleftarrow{F}_x \vdash \overrightarrow{B}$	$\tilde{\mathfrak{M}}_0$	$\overrightarrow{B} + \tilde{F}_x \vdash \tilde{B}$
$\mathfrak{M}_1$	$\overline{P} + \overleftarrow{F}_x \vdash \overleftarrow{W}_x + \overline{P}$	$\tilde{\mathfrak{M}}_1$	$\overline{P} + \tilde{F}_x \vdash \overleftarrow{W}_x + \tilde{P}$
$\mathfrak{E}_0$	$\tilde{B} + \overleftarrow{D} \vdash \overleftarrow{S}$	$\tilde{\mathfrak{E}}_0$	$\tilde{B} + \tilde{D} \vdash \tilde{S}$
$\mathfrak{E}_1$	$\tilde{P} + \overleftarrow{D} \vdash \overleftarrow{S}$	$\tilde{\mathfrak{E}}_1$	$\tilde{P} + \tilde{D} \vdash \tilde{S}$

Figure 14: Local behavior of symbols

3.2. Simulation and catenation

With the previously defined local encoding of CPTS elements, let us proceed by specifying the global encoding and construct catenations embedding the dynamic.

Encoding of a CPTS configuration is made the following way: in the center, the queue is written with symbols $\overline{C}_x$, the upper symbol of the queue being on the right. Right of these elements, the cyclic list of words is repeated infinitely starting from the current word. Words are written with symbols $\overleftarrow{F}_x$ and separated with symbols $\overleftarrow{D}$. The first symbol before the current word is $\overleftarrow{S}$. Furthermore, a symbol is replaced by its altered version where being the last one before a delimiter — i.e., representing the last letter of a word or being a delimiter (or a start symbol) before an empty word. On the left of the queue contents, there is an infinite amount of $\overrightarrow{T}$ symbols.

Proposition 3.2. *For any CPTS evolution, it is possible to construct a catenation scheme embedding the evolution.*

Proof. The catenation uses the symbols presented above. An extract of such a catenation can be found in Figure 16. First, the upper letter of the queue encounters the starting symbol, resulting either on a erasing symbol $\overrightarrow{B}$ (catenation $\mathfrak{S}_0$) or a copying symbol $\overline{P}$ (catenation $\mathfrak{S}_1$) according to the considered letter. This symbol encounters all frozen letters of the word erasing them ($\mathfrak{M}_0$) or transforming them into an unfrozen ones ($\mathfrak{M}_1$). On the last letter, altered catenations ($\tilde{\mathfrak{M}}_0$ or $\tilde{\mathfrak{M}}_1$) alter the symbols which encounter the next

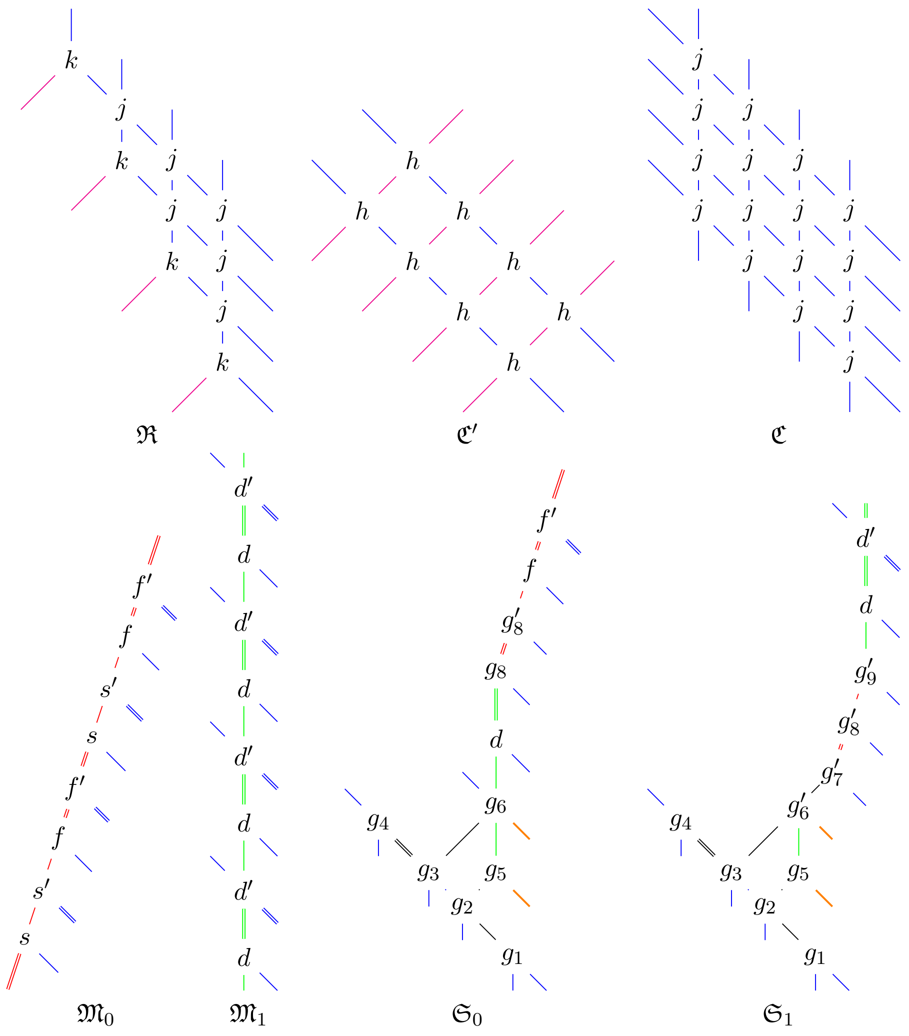


Figure 15: Behaviors of symbols

delimiter, transforming it into a new starting symbol ($\mathfrak{E}_0$ or $\mathfrak{E}_1$). Unfrozen letters are going left, crossing all queue letters ($\mathfrak{C}$). After that, they are added at the end of the queue when colliding with a $\overrightarrow{T}$ symbol (catenation $\mathfrak{R}$). One can note that catenations $\mathfrak{S}_x$ also generate some junk symbols that go left untouched crossing both queue letters (catenation $\tilde{\mathfrak{C}}$) and clock symbols (catenation $\mathfrak{C}$). In the case of an empty word, the behavior is the same up to the fact that the altered symbol of copy or erase is directly generated by an altered start catenation ($\tilde{\mathfrak{S}}_x$). After those steps, the system is ready for a new transition. The simulated system halts when the queue is empty. On the space-time diagram, this condition can be easily expressed by a word indicating that a clock symbol $\overrightarrow{T}$ encounters a start of list symbol $\overleftarrow{S}$. ■

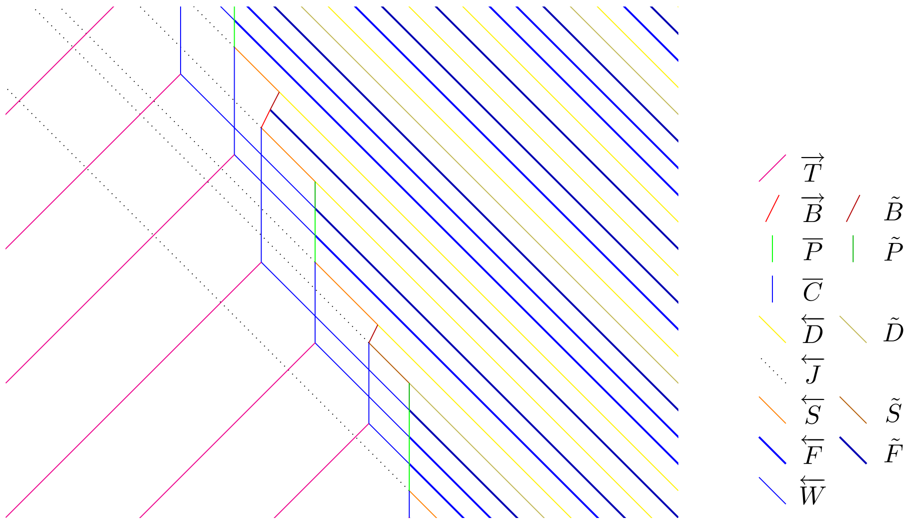


Figure 16: Symbolic behavior of simulation

This explanation conclude the description of dynamic simulation. The last point is to prove that those symbolic behaviors correspond to valid space-time diagrams. At this point, properties of catenation allows us to end the proof without resort to low-level study.

3.3. Validity of simulation

Let us now study the catenation schemes simulating computations of the CPTS. In this section, we shall use properties of catenations to ensure that constructed simulation can really happen in the cellular automaton.

Proposition 3.3. *The previously constructed catenations have all valid affectations. Moreover, constraints on input are independent on the considered evolution.*

Proof. At first, let us look at a global level. Since many symbols are parallel, there are hardly any problem on the order of encounters. The only non-trivial one is that any unfrozen symbols must cross the whole queue before encountering the clock symbol. This implies that is must have crossed the last queue letter before encountering the clock symbols. Since the last queue letter is previous unfrozen letter, the previous condition can be formalized by saying that space between two consecutive clock symbol must be greater than maximum space between two consecutive unfrozen (i.e., copied) letters. At this point, you can see one of the additional conditions on number of consecutive erasing during Turing simulation. With this condition, there is a fixed size for clock spacing ensuring the correct order of collisions independently of the computation.

Now, let us study local constraints. Due to our global approach with collisions and catenations, we have "forgotten" local constraints. In previous proves, the method to ensure these local constraints where by fixing values on the initial configuration and show by induction that they remain consistent. This approach requires a very detailed study and many combinatorial arguments. Here, with the help of catenations, we can have a more global and intuitive approach.

The first remark is that since all our catenations are finite, theorem 1.4 allows us to know affectations such that catenations correspond to real space-time diagram extracts. The set of affectations can be automatically obtained using Presburger arithmetic as described in details in [10].

The only important thing on obtained result is that there exists values for $\mathfrak{S}_0$ and $\mathfrak{S}_1$ that have the same spacing for particles inside $\overleftarrow{S}$ symbol. The same way, it is possible to chose fixed values of spacing for all other signal in symbols that ensure coherence in all catenations. The only exception being the junk symbol which has two possibles values depending on whether it has be generated from a $\mathfrak{S}_0$ or $\mathfrak{S}_1$ catenation.

The last point is to study spacing between symbols. During this process, one look at the catenations formed by the border of the one introduced previously. Even if the used method is the same as previously, some interesting things may be noted: First, the main difference between junk signals and unfrozen letters are the relative positions of particles $\overleftarrow{\imath}$. Another main point encounter when studying the erasing face (see Figure 17). In this face, each erased letter induce a small shift which can not be compensated directly. The solution for this problem is to require that the number of letter is a multiple of 6 which provide a greater and solvable gap. This explains the second restriction introduced in our CPTS. ∎

Conclusion

In this paper, we have shown how any Turing computation can be embedded into rule 110. Starting from a Turing machine, we show how to embed it into a CPTS in Proposition 2.4. Then, we show how to encoding this system into rule 110 space-time diagram (Proposition 3.2) and that this encoding is correct (Proposition 3.3). Thus proving that rule 110 is Turing universal. The main achievement is that the construction can be completely made at high level using particles and collisions which allows to follow at each point what is happening. This construction is very interesting but does only erase particles without creating it, thus having the need to be continually feed with particles. This need of "fuel" is not compatible with intrinsic universality. One open question is whether or not rule 110 is intrinsically universal.

References

[1] N. Boccara, J. Nasser, and M. Roger. Particle like structures and their interactions in spatio temporal patterns generated by one-dimensional deterministic cellular-automaton rules. *Physical Review A*, 44(2):866–875, 1991.

[2] J. Cocke and M. Minsky. Universality of tag systems with p = 2. *Journal of the ACM*, 11(1):15–20, 1964.

[3] M. Cook. Universality in elementary cellular automata. *Complex Systems*, 15:1–40, 2004.

[4] B. Durand and Zs Róka. The game of life: universality revisited. In *Cellular automata (Saissac, 1996)*, pages 51–74. Kluwer Acad. Publ., Dordrecht, 1999.

[5] W. Hordijk, C. R. Shalizi, and J. P. Crutchfield. Upper bound on the products of particle interactions in cellular automata. *Physica D*, 154(3-4):240–258, 2001.

[6] J. Kari. Theory of cellular automata: a survey. *Theoretical Computer Science*, 334:3–33, 2005.

[7] T. Neary and D. Woods. P-completeness of cellular automaton rule. In *ICALP*, Lecture Notes in Computer Science, pages 132–143. Springer, 2006.

[8] N. Ollinger. *Automates cellulaires : structures.* PhD thesis, École Normale Supérieure de Lyon, 2002.

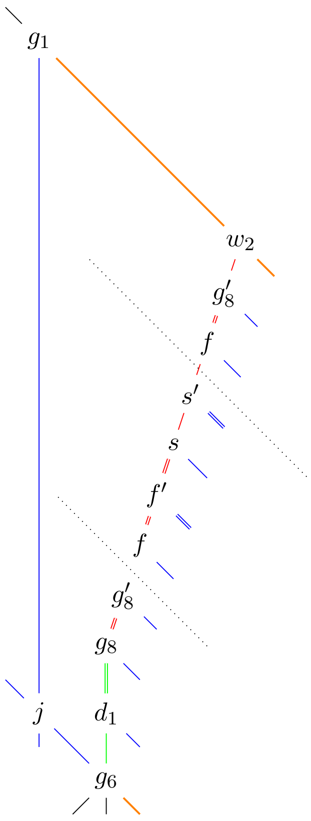


Figure 17: Erasing catenation

[9] N. Ollinger. Universalities in cellular automata; a (short) survey. personal communication *(to appear in JAC 2008)*, 2008.

[10] N. Ollinger and G. Richard. Collisions and their catenations: Ultimately periodic tilings of the plane. (to appear in the proceedings of IFIP-TCS 2008, available at `http://hal.archives-ouvertes.fr/hal-00175397/en/`), 2007.

[11] E. Post. Formal reductions of the general combinatorial decision problem. *American Journal of Mathematics*, 65(2):197–215, 1943.

[12] A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

[13] J. Von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, Ill., 1966.

[14] H. Wang. Tag systems and lag systems. *Mathematische Annalen*, 152(1):65–74, 1963.

[15] S. Wolfram. Universality and complexity in cellular automata. *Physica D. Nonlinear Phenomena*, 10(1-2):1–35, 1984. Cellular automata (Los Alamos, N.M., 1983).

[16] S. Wolfram. *A new kind of science*. Wolfram Media Inc., Champaign, Ilinois, United States, 2002.

[17] D. Woods and T. Neary. On the time complexity of 2-tag systems and small universal Turing machines. *focs*, 0:439–448, 2006.

Journées Automates Cellulaires 2008 (Uzès), pp. 161-173

TWO POINTS OF VIEW TO STUDY THE ITERATES OF A RANDOM CONFIGURATION BY A CELLULAR AUTOMATON

M. SABLIK

Laboratoire d'Analyse, Topologie, Probabilités UMR 6632
Centre de Mathmatiques et Informatique, 39, rue F. Joliot Curie, 13453 Marseille Cedex 13
E-mail address: `sablik@latp.univ-mrs.fr`

Abstract. We study the dynamics of the action of cellular automata on the set of shift-invariant probability measures according two points of view. First, the robustness of the simulation of a cellular automaton on a random configuration can be viewed considering the sensitivity to initial condition in the space of shift-invariant probability measures. Secondly we consider the evolution of the quantity of information in the orbit of a random initial state.

Introduction

Despite the apparent simplicity of their definition, cellular automata can have very complex behaviours which are observed on space time diagrams. To try to understand this complexity, they can be considered as dynamical systems. Generally, one studies the dynamics of the $\mathbb{N}$-action of a cellular automaton on the set of configurations $\mathcal{A}^{\mathbb{Z}}$, where $\mathcal{A}$ is a finite alphabet, endowed with the product topology. However, very simple cellular automata as the power of the shift, denoted σ^m for $m \in \mathbb{Z}$, are sensitive. That is to say, they are considered with a highly chaotic behaviour. This does not correspond to the intuitive idea which give the space-time diagram.

The shortcoming of the Cantor distance is to privilege the central coordinates whereas there may be no reason to give more importance to coordinates around the origin. Moreover, one considers only the action of the cellular automaton without considering the shift action. Indeed, space-time diagrams of a cellular automata $(\mathcal{A}^{\mathbb{Z}}, F)$ are not so different from that of $(\mathcal{A}^{\mathbb{Z}}, \sigma^m \circ F)$ for $m \in \mathbb{Z}$. However, if F is not nilpotent, $\sigma^m \circ F$ is sensitive for m taken quite far from the origin. The reason is that Cantor topology is non-homogeneous, thus a simple transport of information is enough to obtain sensitivity.

One point of view can be to address the $\mathbb{Z} \times \mathbb{N}$-action (σ, F) in order to emphasize the spatiotemporal structure. In [Sab06], one gives general definitions to talk about directional dynamics; the purpose is to study the sets of directions which have a certain kind of dynamics. Another point of view is to kill the $\mathbb{Z}$-action of σ and consider the $\mathbb{N}$-action of F on a σ-invariant object. In this direction, G. Cattaneo, E. Formenti, L. Margara et J. Mazoyer [CFMM97] introduce another topology defined by the Besicovitch pseudo distance

Key words and phrases: Cellular automata, shift invariant topology, entropy of a configuration.

which measures the upper density of the differences between two configurations in order to give the same importance at all cells. For this distance, the shift map is an isometry. However, with this topology, we lose the compactness of the space which is the traditional framework of topological dynamics.

Another natural σ-invariant object is the set of σ-invariant probability measures, denoted $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$. Indeed, a cellular automaton acts on the set of configurations and canonically transforms $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ into another σ-invariant measure denoted $F_*\mu$. Hence, cellular automata also have a natural action on the set of shift-invariant measures. In this space the shift has the same behavior as the identity and a sensitive cellular automaton in this space is not only capable of *"transporting"* information but it is also able to *"create"* new information outwardly.

The study of the action of a cellular automaton on $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ could be interesting when we use cellular automata to simulate. Indeed, generally we observe the action of a cellular automaton on a random initial configuration, that is to say a configuration chosen according to a probability measure $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$. But it is more natural to study the action of a cellular automaton directly on $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ instead the set of configurations chosen according a given probability measure.

In this article we exhibit two problematics:

• The more natural for a dynamical system is the study of sensitivity to initial conditions of the map $F_* : \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}}) \to \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$. This approach can be interesting when we use cellular automata to simulate. Indeed, generally we start the simulation with a random configuration choosen according to a distribution $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$, this approach may help to evaluate the impact of a mistake when we choose the initial configuration with a distribution $\nu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ near to the expected distribution μ.

• The information contained in a random configuration according to a measure $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ can be expressed by the entropy of the shift $h_\mu(\sigma)$. It would be natural to obseve the evolution of this quantity when a cellular automaton acts on μ. It is easy to see that $(h_{F_*^n\mu}(\sigma))_{n\in\mathbb{N}}$ decreases towards $h_\mu^\infty(F)$, but for some cellular automata, like linear cellular automata, it appears a phenomenon of gap between $h_\mu^\infty(F)$ and the apparent entropy $h_\mu^a(F)$, defined by [Mar00], which coresponds to the entropy observed in the central window. This can explain why linear cellular automata have complex space-time diagrams.

We illustrate these two points of view by the study of two important classes: cellular automata with directional equicontinuous points which have very regular space-time diagrams and linear cellular automata which have complex space-time diagrams.

1. Action of a cellular automaton on $\mathcal{A}^{\mathbb{Z}}$ (Background)

1.1. Space of configurations

1.1.1. *Cantor topology.* Let $\mathcal{A}$ be a finite set. We consider $\mathcal{A}^{\mathbb{Z}}$, the *configuration space* of $\mathbb{Z}$-indexed sequences in $\mathcal{A}$. If $\mathcal{A}$ is endowed with the discrete topology, $\mathcal{A}^{\mathbb{Z}}$ is compact, perfect and totally disconnected in the product topology. Moreover one can define a metric on $\mathcal{A}^{\mathbb{Z}}$ compatible with this topology:

$$\forall x, y \in \mathcal{A}^{\mathbb{Z}}, \quad d_C(x,y) = 2^{-\min\{|i| : x_i \neq y_i \ i\in\mathbb{Z}\}}.$$

Let $\mathbb{U} \subset \mathbb{Z}$. For $x \in \mathcal{A}^{\mathbb{Z}}$, the restriction of x to $\mathbb{U}$ is denoted by $x_{\mathbb{U}} \in \mathcal{A}^{\mathbb{U}}$. For a pattern $w \in \mathcal{A}^{\mathbb{U}}$, define $[w]_{\mathbb{U}} = \{x \in \mathcal{A}^{\mathbb{Z}} : x_{\mathbb{U}} = w\}$ the cylinder centered on w.

The shift map $\sigma : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}}$ is defined by $(x_i)_{i\in\mathbb{Z}} \mapsto (x_{i+1})_{i\in\mathbb{Z}}$. A *subshift* $\Sigma \subset \mathcal{A}^{\mathbb{Z}}$ is a closed σ-invariant subset of $\mathcal{A}^{\mathbb{Z}}$. Denote $\mathcal{L}_n(\Sigma)$, the set of patterns $u \in \mathcal{A}^{[0,n-1]}$ such that there exists $x \in \Sigma$ which verifies $x_{[0,n-1]} = u$. The *language* of Σ is $\mathcal{L}(\Sigma) = \cup_{n\in\mathbb{N}}\mathcal{L}_n(\Sigma)$.

1.1.2. *Besicovitch topology.* G. Cattaneo, E. Formenti, L. Margara and J. Mazoyer [CFMM97] introduce the Besicovitch pseudo distance which is σ-invariant. For $x, y \in \mathcal{A}^{\mathbb{Z}}$, it is defined by:

$$d_B(x,y) = \limsup_{n\to\infty} \frac{\mathrm{Card}(\{m \in [-n,n] : x_m \neq y_m\})}{2n+1}.$$

The topology induced by this pseudo-distance has good properties for studying dynamical systems except the compacity.

1.2. Action of a cellular automaton on $\mathcal{A}^{\mathbb{Z}}$

1.2.1. *Definition of CA.* A *cellular automaton* (CA) is a pair $(\mathcal{A}^{\mathbb{Z}}, F)$ where $F : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}}$ is defined by $F(x)_m = \overline{F}((x_{m+u})_{u\in\mathbb{U}})$ for all $x \in \mathcal{A}^{\mathbb{Z}}$ and $m \in \mathbb{Z}$ where $\mathbb{U} \subset \mathbb{Z}$ is a finite set named *neighborhood* and $\overline{F} : \mathcal{A}^{\mathbb{U}} \to \mathcal{A}$ is a *local rule.* By Hedlund's theorem [Hed69], it is equivalent to say that it is a continuous function which commutes with the shift ($\sigma^m \circ F = F \circ \sigma^m$ for all $m \in \mathbb{Z}$).

1.2.2. *General definitions about dynamical systems.* Let (X, d) be a metric space and $F : X \to X$ be a continuous function. There is a lot of definitions to precise the sensitivity to initial conditions of the dynamical system generated by the $\mathbb{N}$-action of F on X. We recall here some of them:

• $x \in X$ is an *equicontinuous point* if for all $\varepsilon > 0$, there exists $\delta > 0$, such that for all $y \in X$, if $d(x,y) < \delta$ then $d(F^n(x), F^n(y)) < \varepsilon$ for all $n \in \mathbb{N}$. Denote $Eq_d(F)$ the set of equicontinuous points. If $x \notin Eq_d(F)$, it is a sensitive point.

• (X, F) is *equicontinuous* if for all $\varepsilon > 0$, there exists $\delta > 0$, such that for all $x, y \in X$, if $d(x,y) < \delta$ then $d(F^n(x), F^n(y)) < \varepsilon$ for all $n \in \mathbb{N}$.

• (X, F) is *sensitive* if there exists $\varepsilon > 0$ such that for all $\delta > 0$ and $x \in X$, there exists $y \in X$ and $n \in \mathbb{N}$ such that $d(x,y) < \delta$ and $d(F^n(x), F^n(y)) > \varepsilon$

• (X, F) is $\mathbb{N}$-*expansive* if there exists $\varepsilon > 0$ such that for all $x \neq y$ there exists $n \in \mathbb{N}$ such that $d(F^n(x), F^n(y)) > \varepsilon$.

In an intuitive sense, sensitivity and expansivity denote a certain complexity of the dynamical system whereas equicontinuity denotes a strong regularity.

1.2.3. *Directional dynamics.* In [Sab06], one adapts the dynamical classification of [Kůr97] according to a direction $\alpha \in \mathbb{R}$. The study of such dynamics make appear some discrete geometry in space time diagrams. We recall here the notion of equicontinuous points. In this case the information is blocked between walls of slope α generated by blocking words.

Proposition 1.1 ([Sab06])**.** *Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be a CA, let $\mathbb{U} = [r, s]$ be a neighborhood of F, and let $\alpha \in \mathbb{R}$. A point $x \in \mathcal{A}^{\mathbb{Z}}$ is* equicontinuous of slope α *if $\forall\varepsilon > 0, \exists\delta > 0$ such that $\forall y \in \mathcal{A}^{\mathbb{Z}}$, if $d_C(x, y) < \delta$ then $d_C(F^n \circ \sigma^{\lfloor \alpha n \rfloor}(x), F^n \circ \sigma^{\lfloor \alpha n \rfloor}(y)) < \varepsilon$ for all $n \in \mathbb{N}$.*

The existence of equicontinuous points is equivalent to the existence of a blocking word u of slope α and width $e \geq \max(\lfloor\alpha\rfloor + 1 + s, -\lfloor\alpha\rfloor + 1 - r)$. *That is to say, there exists $p \in [0, |u| - e]$ such that:*

$$\forall x, y \in [u]_{[0,|u|-1]}, \forall n \in \mathbb{N}, \qquad \sigma^{\lfloor n\alpha \rfloor} \circ F^n(x)_{[p,p+e]} = \sigma^{\lfloor n\alpha \rfloor} \circ F^n(y)_{[p,p+e]}.$$

A CA is *equicontinuous of slope* α if every $x \in \mathcal{A}^{\mathbb{Z}}$ is equicontinuous of slope α.

A CA is $\mathbb{N}$-*expansive of slope* α if there exists $\varepsilon > 0$ such that for all $x, y \in \mathcal{A}^{\mathbb{Z}}$, there exists $n \in \mathbb{N}$ which verifies $d_C(F^n \circ \sigma^{\lfloor \alpha n \rfloor}(x), F^n \circ \sigma^{\lfloor \alpha n \rfloor}(y)) > \varepsilon$.

2. Action of a cellular automaton on $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$

A natural σ-invariant object on which cellular automata can act, is the set of σ-invariant probability measures $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$. This approach is not only theoretical. Indeed, in simulations, the action of a CA is observed on a random configuration chosen according to a probability measure $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$, but it is more easy to study directly the action of the CA on the probability measure μ.

2.1. Measures on $\mathcal{A}^{\mathbb{Z}}$

Let $\mathfrak{B}$ be the Borel sigma-algebra of $\mathcal{A}^{\mathbb{Z}}$. Denote by $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ the set of probability measures on $\mathcal{A}^{\mathbb{Z}}$ defined on the sigma-algebra $\mathfrak{B}$. Usually $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ is endowed with the weak* topology: a sequence $(\mu_n)_{n\in\mathbb{N}}$ of $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ converges to $\mu \in \mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ if and only if for all finite subset $\mathbb{U} \subset \mathbb{Z}$ and for all pattern $u \in \mathcal{A}^{\mathbb{U}}$, one has $\lim_{n\to\infty} \mu_n([u]_{\mathbb{U}}) = \mu([u]_{\mathbb{U}})$. In the weak* topology, the set $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ is compact and metrizable. A metric is defined by:

$$\forall \mu, \nu \in \mathcal{M}(\mathcal{A}^{\mathbb{Z}}), \qquad d_*^{\mathcal{M}}(\mu, \nu) = \sum_{n\in\mathbb{N}} \frac{1}{|\mathcal{A}|^n} \sum_{u \in \mathcal{A}^{[0,n]}} \left|\mu([u]_{[-n,n]}) - \nu([u]_{[-n,n]})\right|.$$

Let $F : \mathcal{A}^{\mathbb{Z}} \to \mathcal{A}^{\mathbb{Z}}$ be a mesurable function. It is possible to consider the action of F on $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ defined by:

$$F_*\mu(B) = \mu(F^{-1}(B)), \text{ for all } \mu \in \mathcal{M}(\mathcal{A}^{\mathbb{Z}}) \text{ and } B \in \mathfrak{B}.$$

A probability measure $\mu \in \mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ is said σ-*invariant* if $\sigma_*\mu = \mu$. Denote $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ the set of σ-invariant probability measures. It is a compact convex subset of $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ (see [DGS76] for more details). A probability measure $\mu \in \mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ is σ-*ergodic* if for all σ-invariant subset $B \in \mathfrak{B}$ (i.e. $\sigma^{-1}(B) = B$ μ-almost everywhere) are trivial (i.e. $\mu(B) = 0$ or 1). The set of σ-ergodic probability measures is denoted by $\mathcal{M}_\sigma^{\text{erg}}(\mathcal{A}^{\mathbb{Z}})$.

Example 2.1. Let $x \in \mathcal{A}^{\mathbb{Z}}$. Define the *Dirac measure in* x by $\delta_x(A) = 1$ if $x \in A$ and 0 if not, where $A \in \mathfrak{B}$. The set of Dirac measures is dense in $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ for the weak* topology.

One remarks that if the configuration is not σ-uniform, the Dirac measure associated is not σ-invariant. However, if we take a σ-periodic configuration x of period $p \in \mathbb{N}$, one constructs a σ-ergodic measure by taking the mean of the Dirac's measures of the σ-orbit: $\widetilde{\delta_x} = \frac{1}{p}\sum_{m\in[0,p-1]} \delta_{\sigma^m(x)}$.

Example 2.2. For all $a \in \mathcal{A}$, put $p_a \in [0,1]$ a real such that $\sum_{a\in\mathcal{A}} p_a = 1$. Define *the Bernoulli measure according to the probability vector* $(p_a)_{a\in\mathcal{A}}$ by $\lambda_{(p_a)_{a\in\mathcal{A}}}([u]_{\mathbb{U}}) = \prod_{m\in\mathbb{U}} p_{u_m}$ for all $u \in \mathcal{L}_{\mathcal{A}^{\mathbb{Z}}}([u]_{\mathbb{U}})$. If all p_a are equal to $\frac{1}{\mathrm{Card}(\mathcal{A})}$, one obtains the *uniform Bernoulli measure* which is just denoted by $\lambda_{\mathcal{A}^{\mathbb{Z}}}$.

2.2. Generic points

For some σ-invariant probability measures, there exist special points of $\mathcal{A}^{\mathbb{Z}}$ which represent the measure. That is to say the frequency of apparition of a pattern corresponds to the measure of the cylinder centered on this pattern. This allows to give a symbolic interpretation of the distance $d_*^{\mathcal{M}}$.

A point $x \in \mathcal{A}^{\mathbb{Z}}$ is *generic* if for all $\mathbb{U} \subset \mathbb{Z}$ finite and for every pattern $u \in \mathcal{A}^{\mathbb{U}}$ the sequence $(f(u,x,n))_{n\in\mathbb{N}}$ converges where

$$f(u,x,n) = \frac{1}{2n+1} \sum_{m\in[-n,n]} \mathbf{1}_{[u]_{\mathbb{U}}}(\sigma^m(x)),$$

is the *frequency of apparition of the pattern u in x at the order n*. The limit of this sequence is denoted $f(u,x)$, this is the *frequency of apparition of the pattern u in x*. Denote $\mathcal{G}$ the set of generic points.

Let $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$. Denote $\mathcal{G}(\mu)$ the set of *generic points of* μ, this is the set of points $x \in \mathcal{G}$ such that for every pattern, the frequency of this pattern in x is equal to the measure of the cylinder centered on this pattern. When μ is σ-ergodic, the Birkhoff Theorem says that $\mu(\mathcal{G}(\mu)) = 1$.

Consider the function $\phi : (\mathcal{G}, d_B) \to (\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}}), d_*^{\mathcal{M}})$ which associates a generic point $x \in \mathcal{G}$ to the measure $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ defined by $\mu([u]_{\mathbb{U}}) = f(u,x)$ for all pattern $u \in \mathcal{A}^{\mathbb{U}}$. It is easy to verify that ϕ is surjective and continuous. Moreover, the image of a generic point by F is also a generic point. Thus the correspondence between $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ and $\mathcal{G}$ is summed up in the following commutative diagram:

$$\begin{tikzcd}
(\mathcal{G}, d_B) \arrow[r, "F"] \arrow[d, "\phi"] & (\mathcal{G}, d_B) \arrow[d, "\phi"] \\
(\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}}), d_*^{\mathcal{M}}) \arrow[r, "F_*"] & (\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}}), d_*^{\mathcal{M}})
\end{tikzcd}$$

2.3. Action of a CA on $\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$

Let $(\mathcal{A}^\mathbb{Z}, F)$ be a CA. The map $\mu \mapsto F_*\mu$ is continuous for the *weak** topology and preserves convex combinations. Thus, $F_* : \mathcal{M}(\mathcal{A}^\mathbb{Z}) \to \mathcal{M}(\mathcal{A}^\mathbb{Z})$ defines a dynamical system. However, for all $x \in \mathcal{A}^\mathbb{Z}$ one has $F_*\delta_x = \delta_{F(x)}$. Thus the map $x \mapsto \delta_x$ allows to consider $(\mathcal{A}^\mathbb{Z}, F)$ as a sub-system of $(\mathcal{M}(\mathcal{A}^\mathbb{Z}), F_*)$, so the dynamics of $F_* : \mathcal{M}(\mathcal{A}^\mathbb{Z}) \to \mathcal{M}(\mathcal{A}^\mathbb{Z})$ contains the dynamics of $F : \mathcal{A}^\mathbb{Z} \to \mathcal{A}^\mathbb{Z}$. Moreover, the weak$*$ topology on $\mathcal{M}(\mathcal{A}^\mathbb{Z})$ privileges the origin. Thus it is preferable to restrict the initial space.

Since F commutes with the shift, if $\mu \in \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$ then $F_*\mu \in \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$. Thus one can study the dynamical system defined by:

$$\begin{array}{rlcll} F_* : & \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}) & \longrightarrow & \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}) & \\ & \mu & \longmapsto & F_*\mu & \quad \text{such that } F_*\mu(B) = \mu(F^{-1}(B)) \ \forall B \in \mathfrak{B}. \end{array}$$

Remark 2.3. If $(\mathcal{A}^\mathbb{Z}, F)$ is a surjective CA, $\lambda_{\mathcal{A}^\mathbb{Z}}$ is F-invariant (see [Hed69]). One deduces that $\lambda_{\mathcal{A}^\mathbb{Z}}$ is a fixed point of F_*.

3. Sensitivity to initial conditions of $F_* : \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}) \to \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$

A natural question in the study of dynamical systems is the sensitivity to initial conditions. This point of view could be interesting when we use cellular automata to simulate in order to characterize distributions which are unstable.

3.1. Expansivity of F_*

The function F_* preserves convex combinations in $\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$. In [BS07], we use this property to show that there are not CA which acts $\mathbb{N}$-expansively on $(\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M})$. This shows that in $\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$, the information cannot be transporting to distinguish initial points.

Theorem 3.1. [BS07] *F_* cannot act $\mathbb{N}$-expansively on $(\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M})$.*

3.2. Equicontinuity

If $(\mathcal{A}^\mathbb{Z}, F)$ is equicontinuous of slope α, according to [Sab06], it is periodic in this direction. Thus it is the same for F_*. One deduces the next proposition.

Proposition 3.2. *Let $(\mathcal{A}^\mathbb{Z}, F)$ be an equicontinuous CA of slope α. Then F_* is equicontinous in $(\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M})$.*

3.3. Equicontinuous points

For some CA it is possible to characterize the set of equicontinuous points. These measures are stable for perturbations. The next example shows that there exist cellular automata with equicontinuous and sensitive points in $(\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M})$.

Example 3.3. Consider the CA defined on $\mathcal{A} = \mathbb{Z}/2\mathbb{Z}$ by $F(x)_i = x_{i-1} \cdot x_i \cdot x_{i+1}$ for all $x \in \mathcal{A}^\mathbb{Z}$ and $i \in \mathbb{Z}$. It is easy to see that for all σ-ergodic probability measure μ which verifies $\mu([0]) > 0$, the sequence $(F_*^n \mu)_{n\in\mathbb{N}}$ converges toward $\delta_{^\infty 0^\infty}$ in $(\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d^\mathcal{M})$. So, one has:

$$Eq_{d_*^\mathcal{M}}(F_*) \cap \mathcal{M}_\sigma^{\mathrm{erg}}(\mathcal{A}^\mathbb{Z}) = \mathcal{M}_\sigma^{\mathrm{erg}}(\mathcal{A}^\mathbb{Z}) \setminus \{\delta_{^\infty 1^\infty}\}.$$

For CA with equicontinuous points of slope α, we characterize a large class of measures which are equicontinuous points in the space $(\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M})$.

Theorem 3.4. *Let $(\mathcal{A}^\mathbb{Z}, F)$ be a CA and let $B \in \mathcal{A}^*$ be a blocking word of slope $\alpha \in \mathbb{R}$. Then every σ-ergodic probability measure $\mu \in \mathcal{M}_\sigma^{erg}(\mathcal{A}^\mathbb{Z})$ such that $\mu([B]) > 0$ is an equicontinuous point of $F_* : (\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M}) \to (\mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z}), d_*^\mathcal{M})$.*

Proof. Let $\varepsilon > 0$, let μ be a σ-ergodic probability measure which charges B and let ν be a σ-invariant measure. For $n \in \mathbb{N}$, one defines $X_{i,n}^k$, the set of points $x \in \mathcal{A}^\mathbb{Z}$ such that there is an occurrence of B in $[-k - \lfloor n\alpha \rfloor, -\lfloor n\alpha \rfloor]$ and another in $[i-1-\lfloor n\alpha \rfloor, k+i-1-\lfloor n\alpha \rfloor]$.

Let i_0 be such that $\sum_{i=i_0+1}^\infty \frac{1}{|\mathcal{A}|^i} \leq \varepsilon$ and let $n \in \mathbb{N}$. Since B is charged by μ, by σ-ergodicity, there exists $k \in \mathbb{N}$ such that $\mu(X_{i,n}^k) \geq 1 - \varepsilon$ for all $i \leq i_0$. Moreover $X_{i,n}^k$ can be written as an union of cylinders centered on $[-k - \lfloor n\alpha \rfloor, k+i-1-\lfloor n\alpha \rfloor]$ of words of $\mathcal{A}^{i+2k}$. By σ-invariance, one deduces that $|\mu(X_{i,n}^k) - \nu(X_{i,n}^k)| \leq |\mathcal{A}|^{i+2k} d(\mu,\nu)$, so:

$$\nu(X_{i,n}^k) \geq 1 - \varepsilon - |\mathcal{A}|^{i+2k} d(\mu,\nu).$$

Let $i \leq i_0$ and let $u \in \mathcal{A}^i$. Put $X_{u,n}^k = F^{-n} \circ \sigma^{-\lfloor \alpha n \rfloor}([u]_{[0,i-1]}) \cap X_{i,n}^k$. Taking the lower bounds of $\mu(X_{i,n}^k)$ and $\nu(X_{i,n}^k)$, one deduces:

$$\begin{aligned}|F_*^n\mu([u]_{[0,i-1]}) - F_*^n\nu([u]_{[0,i-1]})| &\leq |F_*^n\mu([u]_{[0,i-1]}) - \mu(X_{u,n}^k)| + |\mu(X_{u,n}^k) - \nu(X_{u,n}^k)| \\ &\quad + |F_*^n\nu([u]_{[0,i-1]}) - \nu(X_{u,n}^k)| \\ &\leq \varepsilon + |\mu(X_{u,n}^k) - \nu(X_{u,n}^k)| + \varepsilon + |\mathcal{A}|^{|u|+2k} d(\mu,\nu).\end{aligned}$$

A summation gives for all $i \leq i_0$ the following inequality:

$$\begin{aligned}\sum_{u\in\mathcal{A}^i} |F_*^n\mu([u]_{[0,i-1]}) - F_*^n\nu([u]_{[0,i-1]})| &\leq 2\varepsilon|\mathcal{A}|^i + |\mathcal{A}|^{2i+2k} d(\mu,\nu) \\ &\quad + \sum_{u\in\mathcal{A}^i} |\mu(X_{u,n}^k) - \nu(X_{u,n}^k)|.\end{aligned}$$

Let $Y_{u,n}^k$ be the set of words $v \in \mathcal{A}^{|u|+2k}$ such that there exists $y \in F^{-n} \circ \sigma^{-\lfloor \alpha n \rfloor}[u]_{[0,i-1]} \cap [v]_{[-k-\lfloor n\alpha \rfloor, k+|u|-\lfloor n\alpha \rfloor]} \cap X_{|u|,n}^k$. Since B is a blocking word of slope α, for all $v \in Y_{u,n}^k$, for all $x \in [v]_{[-k-\lfloor n\alpha \rfloor, k+|u|-\lfloor n\alpha \rfloor]}$, one has $F^n(x) \circ \sigma^{\lfloor \alpha n \rfloor}(x)_{[0,|u|-1]} = u$. One deduces that:

$$X_{u,n}^k = F^{-n} \circ \sigma^{-\lfloor \alpha n \rfloor}([u]_{[0,i-1]}) \cap X_{|u|,n}^k = \bigcup_{Y_{u,n}^k} [v]_{[-k-\lfloor n\alpha \rfloor, k+|u|-\lfloor n\alpha \rfloor]}.$$

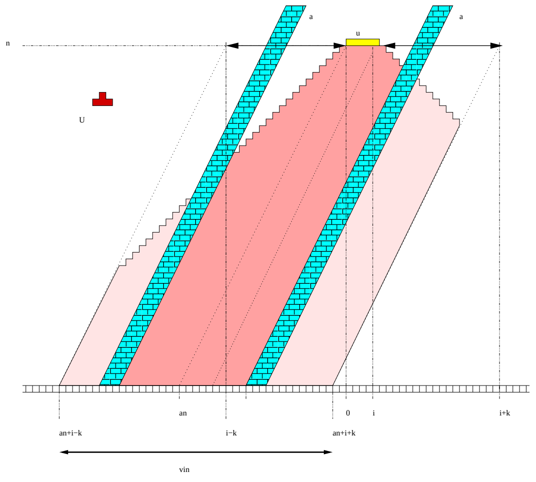


Figure 1: Blocking word of slope α and $d_*^{\mathcal{M}}$-equicontinuity.

Thus, it is possible to decompose the sets $X_{u,n}^k$ to obtain a sum of measures of cylinders centered on $\mathbb{V} = [-k - \lfloor n\alpha \rfloor, k + |u| - \lfloor n\alpha \rfloor]$ with words in $\mathcal{A}^{i+2k}$:

$$\begin{aligned}
\sum_{u\in\mathcal{A}^i} |\mu(X_{u,n}^k) - \nu(X_{u,n}^k)| &= \sum_{u\in A^i} \Big| \sum_{v\in Y_{u,n}^k} \mu([v]_{\mathbb{V}}) - \nu([v]_{\mathbb{V}}) \Big| \\
&\leq \sum_{v\in\mathcal{A}^{i+2k}} |\mu([v]_{[0,i-1]}) - \nu([v]_{[0,i-1]})|.
\end{aligned}$$

By summation of previous inequalities, it follows that:

$$\begin{aligned}
d_*^{\mathcal{M}}(F_*^n\mu, F_*^n\nu) &= \sum_{i\leq i_0} \frac{1}{|\mathcal{A}|^{2i}} \sum_{u\in\mathcal{A}^i} |F_*^n\mu([u]_{[0,i-1]}) - F_*^n\nu([u]_{[0,i-1]})| \\
&\quad + \sum_{i>i_0} \frac{1}{|\mathcal{A}|^{2i}} \sum_{u\in\mathcal{A}^i} |F_*^n\mu([u]_{[0,i-1]}) - F_*^n\nu([u]_{[0,i-1]})| \\
&\leq \frac{2\varepsilon}{|\mathcal{A}|-1} + |\mathcal{A}|^{2k}(1+|\mathcal{A}|^{i_0})d(\mu,\nu) + \varepsilon.
\end{aligned}$$

This shows that the orbits $(F_*^n\mu)_{n\in\mathbb{N}}$ and $(F_*^n\nu)_{n\in\mathbb{N}}$ stay close to each other when μ and ν are close enough. ■

3.4. The case of algebraic CA

The uniform Bernoulli measure (see example 2.2) has an important role in the study of σ-invariant measures. It is known that for a large class of algebraic CA and a large class of measures, the Cesàro mean of the iterates of a measure by the CA converges to the uniform Bernoulli measure. This result was proved with tools from stochastic processes in [FMMN00], and with harmonic analysis tools in [PY02] and [PY04]. We use this result to show the $d_*^{\mathcal{M}}$-sensitivity of linear CA.

Theorem 3.5. *Let $\mathcal{A}$ be an Abelian finite group and let $(\mathcal{A}^{\mathbb{Z}}, F)$ be a non trivial linear CA (that is to say a group endomorphism on the product group $\mathcal{A}^{\mathbb{Z}}$ which is not a product of shifts). Then $Eq_{d_*^{\mathcal{M}}}(F_*) = \emptyset$.*

Proof. Since $(\mathcal{A}^{\mathbb{Z}}, F)$ is a non trivial linear CA, there exist p prime, a surjective endomorphism $\pi : \mathcal{A} \to \mathbb{Z}/p\mathbb{Z}$ and a factor CA $(\mathbb{Z}/p\mathbb{Z}, \widehat{F})$ such that $\pi \circ F = \widehat{F} \circ \pi$ (where π is extended coordinate to coordinate). It is easy to verify that $\widehat{F}$ is a non-trivial linear CA on $\mathbb{Z}/p\mathbb{Z}$. Since π_* is open, it is sufficient to prove the sensitivity for this factor of F_*. Thus we assume that we are in this case.

Let $((\mathbb{Z}/p\mathbb{Z})^{\mathbb{Z}}, F)$ be a nontrivial linear CA with p prime. In [PY02], they show that there is a weak* dense set in $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$, the harmonic measures set denoted $\mathcal{H}$, such that every measure $\mu \in \mathcal{H}$ verifies

$$\lim_{n \in \mathbb{J} \to \infty} d_*^{\mathcal{M}}(F_*^n \mu, \lambda_{\mathcal{A}^{\mathbb{Z}}}) = 0,$$

where $\lambda_{\mathcal{A}^{\mathbb{Z}}}$ is the uniform Bernoulli measure and $\mathbb{J}$ is a subset of $\mathbb{N}$ of upper density 1.

Let P be the set of (σ, F)-periodic points, it is a dense subset of $\mathcal{A}^{\mathbb{Z}}$ [BK99]. According to example 2.1, it is easy to see that the set $\{\widetilde{\delta_x} : x \in P\}$ is weak* dense in $\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$.

Let $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ such that $\mu \neq \lambda_{\mathcal{A}^{\mathbb{Z}}}$ and let $\varepsilon < \frac{1}{2} d_*^{\mathcal{M}}(\lambda_{\mathcal{A}^{\mathbb{Z}}}, \mu)$. For all $\delta < \varepsilon$. There exists $\mu' \in \mathcal{H}$ and $x \in P$ such that $d_*^{\mathcal{M}}(\mu, \mu') < \delta$ and $d_*^{\mathcal{M}}(\mu, \widetilde{\delta_x}) < \delta$. Thus one has $\lim_{n \in \mathbb{J} \to \infty} d_*^{\mathcal{M}}(F_*^n \mu', \lambda_{\mathcal{A}^{\mathbb{Z}}}) = 0$, where $\mathbb{J}$ is a subset of $\mathbb{N}$ of density 1. Moreover, if p is the (σ, F)-period of x, one has $F_*^{pn} \widetilde{\delta_x} = \widetilde{\delta_x}$ for all $n \in \mathbb{N}$. Since $\mathbb{J}$ has upper density 1, there exists $n \in \mathbb{N}$ such that $pn \in \mathbb{J}$ and $d_*^{\mathcal{M}}(F_*^{pn} \mu', \lambda_{\mathcal{A}^{\mathbb{Z}}}) < \varepsilon$. Thus one has $d_*^{\mathcal{M}}(F_*^{pn} \widetilde{\delta_x}, F_*^{pn} \mu') > \varepsilon$. One deduces that $\mu \notin Eq_{d_*^{\mathcal{M}}}(F_*)$.

If $\mu = \lambda_{\mathcal{A}^{\mathbb{Z}}}$ there exists μ' such that the sequence $(F_*^n \mu')_{n \in \mathcal{A}^{\mathbb{Z}}}$ has at least two adherence points: one of them is $\mu = \lambda_{\mathcal{A}^{\mathbb{Z}}}$ [PY02], denote μ'' another. Since $\mu = \lambda_{\mathcal{A}^{\mathbb{Z}}}$ is F_*-invariant, it is possible to find in the sequence $(F_*^n \mu')_{n \in \mathcal{A}^{\mathbb{Z}}}$ a point close to μ but μ'' is a closure point of the orbit of this point. So $\mu \notin Eq_{d_*^{\mathcal{M}}}(F_*)$.

Thus $Eq_{d_*^{\mathcal{M}}}(F_*) = \emptyset$, but this method do not allow to obtain an uniform sensitive constant. There is a problem around of $\lambda_{\mathcal{A}^{\mathbb{Z}}}$. ■

4. Quantity of information in a μ-generic configuration

4.1. Problematic

Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be a CA. A configuration $x \in \mathcal{A}^{\mathbb{Z}}$ converges generally to the *limit set* $\Lambda_F = \bigcap_{n \in \mathbb{N}} F^n(\mathcal{A}^{\mathbb{Z}})$. However, when you look the simulation of a CA on points chosen according to a σ-invariant probability measure $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$, the limit set capture so many points. This does not correspond to the observation. That is why P. Kůrka and A. Maass [KM00] introduce the concept of *μ-limit set*, denoted $\Lambda_F(\mu)$, defined by:

$$u \notin \mathcal{L}(\Lambda_F(\mu)) \Longleftrightarrow \lim_{n \to \infty} F_*([u]_{[0,|u|-1]}) = 0.$$

Naturally, one has $\Lambda_F(\mu) \subset \Lambda_F$. In symbolic dynamics, the complexity of a subshift Σ is mesured thanks to the topological entropy:

$$h_{\text{top}}(\Sigma) = \lim_{n \to \infty} \frac{\log(\text{Card}(\mathcal{L}_n(\Sigma)))}{n}.$$

Let $\mu \in \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$. It is also possible to define the metric entropy of σ relative to μ by:

$$h_\mu(\sigma) = \lim_{n\to\infty} -\frac{1}{n}\sum_{u\in\mathcal{A}^n} \mu([u])\log(\mu([u])).$$

We refer to [DGS76] for more general definitions and main properties.

In fact, the information contained in a μ-generic configuration can be expressed by the entropy of the shift. A comparative study of the entropy of the shift and the Kolmogorov complexity was carried out by A. Brudno [Bru82]. It could be interesting to study the sequence $(h_{F_*^n\mu}(\sigma))_{n\in\mathbb{N}}$ which corresponds to the evolution of the quantity of information when we start with a random configuration chosen according to μ.

Let $\mathbb{U} \subset \mathbb{Z}$ be finite, put $\mathcal{P}_\mathbb{U} = \{[u]_\mathbb{U} : u \in \mathcal{A}^\mathbb{U}\}$. The *quantity of information* in the partition $\mathcal{P}_\mathbb{U}$ is defined by $H_{F_*^n\mu}(\mathcal{P}_\mathbb{U}) = -\sum_{u\in\mathcal{A}^\mathbb{U}} F_*^n\mu([u]_\mathbb{U})\log(F_*^n\mu([u]_\mathbb{U}))$.

To symplify the notation, denote $\mathcal{P}_k = \mathcal{P}_{[0,k-1]}$, thus $h_{F_*^n\mu}(\sigma) = \lim_{k\to\infty}\frac{1}{k}H_{F_*^n\mu}(\mathcal{P}_k)$ corresponds to the metric entropy after n iterations of F. If $\mathbb{U} = [r,s]$ is a neighborhood of F, it is easy to see that $H_{F_*^{n+1}\mu}(\mathcal{P}_{0,k-1}) \leq H_{F_*^n\mu}(\mathcal{P}_{[r,s+k-1]})$. One deduces:

Proposition 4.1. *Let $(\mathcal{A}^\mathbb{Z}, F)$ be a CA and $\mu \in \mathcal{M}_\sigma(\mathcal{A}^\mathbb{Z})$. If $n_1 \leq n_2$ then $h_{F_*^{n_1}\mu}(\sigma) \geq h_{F_*^{n_2}\mu}(\sigma)$.*

Thus the quantity of information in a random configuration decreases under the action of a CA. It is natural since a CA does not create explicitly information. One deduces that $(h_{F_*^n\mu}(\sigma))_{n\in\mathbb{N}}$ decrease, so it is possible to define the *infinite entropy* of $(\mathcal{A}^\mathbb{Z}, F)$ by:

$$h_\mu^\infty(F) = \lim_{n\to\infty} h_{F_*^n\mu}(\sigma) = \inf_{n\in\mathbb{N}} h_{F_*^n\mu}(\sigma) = \lim_{n\to\infty}\lim_{k\to\infty}\frac{1}{k}H_{F_*^n\mu}(\mathcal{P}_k).$$

One problem of this definition is that at each time, we consider arbitrary long patterns: we take the limit according to the parameter k before the time parameter n. So it is difficult to detect the correlations between paterns of the same length under the action of F. However, if we observe the evolution of a CA, we just look a fixed window of the space-time diagram. Thus, the entropy observed would be naturally defined as the inversion of limits in the formula of $h_\mu^\infty(F)$. That is why B.Martin defines in [Mar00] *the apparent entropy*:

$$h_\mu^a(F) = \lim_{k\to\infty}\limsup_{n\to\infty}\frac{1}{k}H_{F_*^n\mu}(\mathcal{P}_k).$$

In [Mar00], the author exhibits the link between $h_\mu^a(F)$ and the rate of compression by gzip of space-time diagrams where the initial configuration are μ-generic configurations.

4.2. Problem of gap between $h_\mu^\infty(F)$ and $h_\mu^a(F)$

We propose to exhibit links between the different notions of complexity: $h_\mu^\infty(F)$, $h_\mu^a(F)$ and $h_{\mathrm{top}}(\Lambda_F(\mu))$. Theorem 4.2 shows a natural inequality between the different values, in particular it can appear a phenomenon of gap between $h_\mu^\infty(F)$ and $h_\mu^a(F)$. This means that for a CA F and an initial distribution μ, the complexity which is observed in the space-time diagram (i.e. $h_\mu^a(F)$) is different from the expected value (i.e. $h_\mu^\infty(F)$). Theorem 4.3 proves that for each CA there exists a dististribution such that the complexity observed in the space time diagram corresponds to the disorder spreaded by the CA.

Theorem 4.2. *Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be a CA and $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$, one has:*

$$h_{top}(\Lambda_F) \geq h_{top}(\Lambda_F(\mu)) \geq h^a_\mu(F) \geq \sup_{\nu \in \mathcal{V}} h_\nu(\sigma) \geq \inf_{\nu \in \mathcal{V}} h_\nu(\sigma) \geq h^\infty_\mu(F),$$

where $\mathcal{V}$ is the set of closure points of $(F^n_* \mu)_{n \in \mathbb{N}}$.

Proof. Let k and n in $\mathbb{N}$, one has:

$$\begin{aligned} H_{F^n_*\mu}(\mathcal{P}_k) &= - \sum_{u \in \mathcal{L}_k(\mathcal{A}^{\mathbb{Z}})} F^n_*\mu([u]_{[0,k-1]}) \log(F^n_*\mu([u]_{[0,k-1]})) \\ &= - \sum_{u \in \mathcal{L}_k(\Lambda_F(\mu))} F^n_*\mu([u]_{[0,k-1]}) \log(F^n_*\mu([u]_{[0,k-1]})) \\ &\quad - \sum_{u \in \mathcal{L}_k(\mathcal{A}^{\mathbb{Z}}) \setminus \mathcal{L}(\Lambda_F(\mu))} F^n_*\mu([u]_{[0,k-1]}) \log(F^n_*\mu([u]_{[0,k-1]})). \end{aligned}$$

By definition of $\Lambda_F(\mu)$, one has $\sum_{u \in \mathcal{L}_k(\mathcal{A}^{\mathbb{Z}}) \setminus \mathcal{L}(\Lambda_F(\mu))} F^n_*\mu([u]_{[0,k-1]}) \log(F^n_*\mu([u]_{[0,k-1]})) \to 0$ when $n \to 0$. We deduce that $h_{\text{top}}(\Lambda_F(\mu)) \geq h^a_\mu(F)$.

Let $\nu \in \mathcal{V}$ be a limit of a subsequence $(F^{n_i}\mu)_{i \in \mathbb{N}}$ (there exist such subsequences since $(\mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}}), d^{\mathcal{M}}_*)$ is compact). By continuity of $\mu \mapsto H_\mu(\mathcal{P}_k)$, one has:

$$\limsup_{n \to \infty} H_{F^n_*\mu}(\mathcal{P}_k) \geq \limsup_{i \to \infty} H_{F^{n_i}\mu}(\mathcal{P}_k) = H_\nu(\mathcal{P}_k).$$

So one obtains $h^a_\mu(F) \geq h_\nu(\sigma)$.

Let $\nu \in \mathcal{V}$ be a limit of a subsequence $(F^{n_i}\mu)_{i \in \mathbb{N}}$, by upper semi-continuity of $\mu \mapsto h_\mu(\sigma)$ (see [DGS76]), one has $h_\nu(X, \sigma) \geq \limsup_{i \to \infty} h_{F^{n_i}\mu}(X, \sigma) = h^\infty_\mu(F)$. ■

Theorem 4.3. *Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be a CA. There exists $\mu \in \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}})$ such that:*

$$h^\infty_\mu(F) = h^a_\mu(F) = h_{top}(\Lambda_F(\mu)) = h_{top}(\Lambda(F)).$$

Proof. Let $n \in \mathbb{N}$, since the subshift $F^n(\mathcal{A}^{\mathbb{Z}})$ is intrinsically ergodic (that is to say, there exists an unique σ-ergodic measure of maximal entropy, see [DGS76]), there exists ν_n such that $h_{\nu_n}(\sigma) = h_{\text{top}}(F^n(\mathcal{A}^{\mathbb{Z}}))$. Moreover the operator $F^n_* : \mathcal{M}_\sigma(\mathcal{A}^{\mathbb{Z}}) \to \mathcal{M}_\sigma(F^n(\mathcal{A}^{\mathbb{Z}}))$ defined by $\mu \mapsto F^n_*\mu$ is surjective, thus there exists $\mu_n \in \mathcal{M}(\mathcal{A}^{\mathbb{Z}})$ such that $\nu_n = F^n_*\mu_n$ for all $n \in \mathbb{N}$.

Let μ be a limit of a subsequence $(\mu_{n_i})_{i \in \mathbb{N}}$ of $(\mu_n)_{n \in \mathbb{N}}$. Let $\varepsilon > 0$, there exists $n \in \mathbb{N}$ such that $h^\infty_\mu(F) + \varepsilon \geq h_{F^n_*\mu}(\sigma)$. Moreover by upper semi-continuity of the entropy, there exists $i_0 \in \mathbb{N}$ such that $h_{F^n_*\mu}(\sigma) \geq h_{F^n_*\mu_{n_i}}(\sigma) - \varepsilon$ for all $i \geq i_0$. So if we choose i such that $n_i \geq n$ one has:

$$h^\infty_\mu(F) + 2\varepsilon \geq h_{F^n_*\mu_{n_i}}(\sigma) \geq h_{F^{n_i}_*\mu_{n_i}}(\sigma) = h_{\text{top}}(F^{n_i}(\mathcal{A}^{\mathbb{Z}})) \geq h_{\text{top}}(\Lambda(F)).$$

The inequality is true for every $\varepsilon > 0$, we deduce that $h^\infty_\mu(F) \geq h_{\text{top}}(\Lambda(F))$. The equality follows by the previous theorem. ■

Remark 4.4. If $(\mathcal{A}^{\mathbb{Z}}, F)$ is surjective, $\lambda_{\mathcal{A}^{\mathbb{Z}}}$ is a fixed point of F_*, but it is also the unique measure of maximal entropy, so $\Lambda_F(\lambda_{\mathcal{A}^{\mathbb{Z}}}) = \mathcal{A}^{\mathbb{Z}}$. The measure $\lambda_{\mathcal{A}^{\mathbb{Z}}}$ verifies the case of equality in Theorem 4.2.

4.3. Illustration for some class of cellular automata

By Theorem 4.3, for all CA there exists a measure μ such that $h^{\infty}_{\mu}(F) = h^{a}_{\mu}(F)$. In this subsection we search to link between the dynamic of a cellular automaton and the case of equality in Theorem 4.2.

Proposition 4.5. *Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be an equicontinuous CA of slope α and $\mu \in \mathcal{M}_{\sigma}(\mathcal{A}^{\mathbb{Z}})$. Then there exists $m \in \mathbb{N}$ such that $h^{\infty}_{\mu}(F) = h^{a}_{\mu}(F) = h_{F^m_*\mu}(\sigma)$.*

Proof. By Theorem 4.3 of [Sab06], there exist a period $p \in \mathbb{N}$ and a preperiod $m \in \mathbb{N}$ such that $\sigma^{\lfloor (m+p)\alpha \rfloor} \circ F^{m+p}(x) = \sigma^{\lfloor m\alpha \rfloor} \circ F^{m}$. One deduces that $\mathcal{V}$, in Theorem 4.2, is $\{F^{m+n}_{*}\mu : n \in [0, p-1]\}$. Thus $h^{\infty}_{\mu}(F) = h^{a}_{\mu}(F)$. ■

Proposition 4.6. *Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be a CA which has two blocking word B' and B'' of slope respectively α' and α''. Let $\mu \in \mathcal{M}^{erg}_{\sigma}(\mathcal{A}^{\mathbb{Z}})$ such that $\mu(B') > 0$ and $\mu(B'') > 0$. Then $h^{\infty}_{\mu}(F) = h^{a}_{\mu}(F) = h_{top}(\Lambda_F(\mu)) = 0$.*

Proof. Since μ charges two blocking words of different slope, $(\mathcal{A}^{\mathbb{Z}}, F)$ has two directions of μ-almost equicontinuity. According to Theorem 4.8 of [Sab06], one deduces that there exists $\mathcal{A}_{\infty} \subset \mathcal{A}$ such that $\Lambda_F(\mu) = \{{}^{\infty}a^{\infty} : a \in \mathcal{A}_{\infty}\}$. One deduces the result. ■

Proposition 4.7. *Let $(\mathcal{A}^{\mathbb{Z}}, F)$ be an algebraic CA. There is a dense set of measures $\mathcal{H}$ such that for all $\mu \in \mathcal{H}$ one has $h^{\infty}_{\mu}(F) \neq h^{a}_{\mu}(F) = \log(\mathrm{Card}(\mathcal{A}))$.*

Proof. In [PY02], they show that there is a weak* dense set in $\mathcal{M}(\mathcal{A}^{\mathbb{Z}})$, the harmonic measures set denoted $\mathcal{H}$, such that every measure $\mu \in \mathcal{H}$ verifies $\lim_{n\in\mathbb{J}\to\infty} d^{\mathcal{M}}_{*}(F^n_*\mu, \lambda_{\mathcal{A}^{\mathbb{Z}}}) = 0$, where $\lambda_{\mathcal{A}^{\mathbb{Z}}}$ is the uniform Bernoulli measure and $\mathbb{J}$ is a subset of $\mathbb{N}$ of upper density 1. By Theorem 4.2, one deduces that $h^{a}_{\mu}(F) = h_{\lambda_{\mathcal{A}^{\mathbb{Z}}}}(\sigma) = \log(\mathrm{Card}(\mathcal{A}))$. However, if $\mu \neq \lambda_{\mathcal{A}^{\mathbb{Z}}}$, one has $h^{\infty}_{\mu}(F) \leq h_{\mu}(\sigma) < \log(\mathrm{Card}(\mathcal{A}))$. ■

Thus there is a phenomenon of gap between $h^{\infty}_{\mu}(F)$ and $h^{a}_{\mu}(F)$. This means that the space-time diagram of algebraic CA looks more complex than the initial configuration. In fact, the combinatory involved by the local rule is so important and it seems to appear information. On the contrary, when $h^{\infty}_{\mu}(F) = h^{a}_{\mu}(F)$, this means that the CA cannot mix suffisently the informations contained in a μ-random configuration.

References

[BK99] Mike Boyle and Bruce Kitchens. Periodic points for onto cellular automata. *Indag. Math. (N.S.)*, 10(4):483–493, 1999.

[Bru82] A. A. Brudno. Entropy and the complexity of the trajectories of a dynamic system. *Trudy Moskov. Mat. Obshch.*, 44:124–149, 1982.

[BS07] Laurent Bienvenu and Mathieu Sablik. The dynamics of cellular automata in shift-invariant topologies. *In Proc. of Development in Language Theory*, 2007.

[CFMM97] Gianpiero Cattaneo, Enrico Formenti, Luciano Margara, and Jacques Mazoyer. A shift-invariant metric on $S^{\mathbf{Z}}$ inducing a non-trivial topology. In *Mathematical foundations of computer science 1997 (Bratislava)*, volume 1295 of *Lecture Notes in Comput. Sci.*, pages 179–188. Springer, Berlin, 1997.

[DGS76] Manfred Denker, Christian Grillenberger, and Karl Sigmund. *Ergodic theory on compact spaces.* Springer-Verlag, Berlin, 1976. Lecture Notes in Mathematics, Vol. 527.

[FMMN00] Pablo A. Ferrari, Alejandro Maass, Servet Martínez, and Peter Ney. Cesàro mean distribution of group automata starting from measures with summable decay. *Ergodic Theory Dynam. Systems*, 20(6):1657–1670, 2000.

[Hed69] Gustav A. Hedlund. Endormorphisms and automorphisms of the shift dynamical system. *Math. Systems Theory*, 3:320–375, 1969.

[KM00] Petr Kůrka and Alejandro Maass. Limit sets of cellular automata associated to probability measures. *J. Statist. Phys.*, 100(5-6):1031–1047, 2000.

[Kůr97] Petr Kůrka. Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory Dynam. Systems*, 17(2):417–433, 1997.

[Mar00] Bruno Martin. Apparent entropy of cellular automata. *Complex Systems*, 12(2):135–155, 2000.

[PY02] Marcus Pivato and Reem Yassawi. Limit measures for affine cellular automata. *Ergodic Theory Dynam. Systems*, 22(4):1269–1287, 2002.

[PY04] Marcus Pivato and Reem Yassawi. Limit measures for affine cellular automata. II. *Ergodic Theory Dynam. Systems*, 24(6):1961–1980, 2004.

[Sab06] Mathieu Sablik. Directional dynamics for cellular automata: A sensitivity to initial condition approach. to appear in *Theoretical Computer Science*, September 2006.

Journées Automates Cellulaires 2008 (Uzès), pp. 174-181

TWO-DIMENSIONAL CELLULAR AUTOMATA RECOGNIZER EQUIPPED WITH A PATH

VÉRONIQUE TERRIER

GREYC, Campus II, Université de Caen, F-14032 Caen Cedex, France

Abstract. In this paper, two-dimensional cellular automata as one-dimensional language recognizers are considered. Following the approach of M. Delorme and J. Mazoyer to embed one-dimensional words into two-dimensional array, we deal with two-dimensional cellular automata equipped with a path. In this context, we investigate regular language recognition.

1. Introduction

Cellular automata (CA) is a major model of massively parallel computation. Famous examples [1, 3] illustrate the ability of CA to distribute and synchronize the information in a very efficient way. However, to determine to which extent CA can fasten sequential computation is not simple. In this context, a lot of interest has been devoted to evaluate the computation ability of one-dimensional CA as language recognizer. A question is whether increasing the dimension allows to increase the computation ability. Actually, the combinatorial capabilities of CA become more complex and new problems arise. The first ambiguity is in the way the one-dimensional input words are fed into higher dimensional arrays. For two-dimensional array, several manners have been considered. In this paper, we follow the approach introduced by M. Delorme and J. Mazoyer in [2]. They embed the input words along a path coded in an additional layer of the two-dimensional array. They prove that such CA equipped with an Archimedian path or an Hilbert path are able to recognize in real time regular languages. Here we get rid of some obstacles in considering von Neumann neighborhood instead of Moore neighborhood and especially in assuming that all cells know the position of the output. In this context, we present below such CA equipped with a path which recognize regular languages in real time whatever the path is like. The basic features of the algorithm exploit shrinking techniques.

Key words and phrases: Two-dimensional cellular automata, language recognition, regular languages.

2. Definitions

First we recall the definitions. They are mainly following the ones introduced in [2].

Definition 2.1. A deterministic finite automaton (FA) is specified by a quintuplet $(\Sigma, Q, \delta, q_{init}, Q_{\text{accept}})$ where Σ represents the input alphabet, Q the finite set of states, $q_{init} \in Q$ the initial state, $Q_{\text{accept}} \subset Q$ the set of accepting states and $\delta : Q \times \Sigma \mapsto Q$ the transition function.
Extending the transition function to strings over Σ^*, the language recognized by a finite automaton $\mathcal{F} = (\Sigma, Q, \delta, q_{init}, Q_{\text{accept}})$ is defined as $\mathcal{L}(\mathcal{F}) = \{w \in \Sigma : \delta(q_{init}, w) \in Q_{\text{accept}}\}$.

Definition 2.2. A two-dimensional cellular automaton is a two-dimensional array of identical finite automata (cells) indexed by $\mathbb{Z}^2$. In vector notation each cell is identified to its vector position $\mathbf{c} = (x_{\mathbf{c}}, y_{\mathbf{c}})$. The communication links are finite and uniform for every cell. They are specified by a finite subset of $\mathbb{Z}^2$ called the neighborhood. Each cell takes on a value from a finite set, the set of states. All cells evolve synchronously at discrete time steps according to the states of their local neighborhood. Formally the behavior of a two-dimensional cellular automaton is specified by a triplet (S, V, δ) where S represents the set of states, $V \subset \mathbb{Z}^2$ the neighborhood of size k, $\delta : S^k \mapsto S$ the transition function. Here we will restrict the neighborhood to the von Neumann one (of size 5): $V = \{\mathbf{v} \in \mathbb{Z}^2 : |x_{\mathbf{v}} + y_{\mathbf{v}}| \leq 1\}$.

Definition 2.3. The vector notations $\mathbf{n}, \mathbf{e}, \mathbf{s}, \mathbf{w}$ will denote the four directions : the north $\mathbf{n} = (-1, 0)$, the east $\mathbf{e} = (0, 1)$, the south $\mathbf{s} = (1, 0)$ and the west $\mathbf{w} = (0, -1)$. A four-connected oriented path is any sequence of positions $\mathbf{c_1}, \cdots, \mathbf{c_n}$ in $\mathbb{Z}^2$ where any two successive positions are four-adjacent: $\mathbf{c_{i+1}} - \mathbf{c_i} \in \{\mathbf{n}, \mathbf{e}, \mathbf{s}, \mathbf{w}\}$. In addition, we will restrict to simple path. That means that repeated positions are forbidden: if $i \neq j$ then $\mathbf{c_i} \neq \mathbf{c_j}$. In particular, the path is non-looping. In the sequel, we will refer to a simple and four-connected oriented path as simply a path.

Definition 2.4. A CA equipped with a path p is a CA with an additional layer which records the path p. Formally, it is specified by a quadruplet (P, S, V, δ) with $P = \{\mathbf{n}, \mathbf{s}, \mathbf{e}, \mathbf{w}, \sharp\} \times \{\mathbf{n}, \mathbf{s}, \mathbf{e}, \mathbf{w}, \sharp\} \setminus \{(\mathbf{n}, \mathbf{s}), (\mathbf{s}, \mathbf{n}), (\mathbf{e}, \mathbf{w}), (\mathbf{w}, \mathbf{e})\}$ the set of the symbols recording the path, S the set of the states, V the von Neumann neighborhood and $\delta : (S \times P)^5 \mapsto S$ the transition function. Given $p = (\mathbf{c_1}, \cdots, \mathbf{c_n})$, on the additional layer, the symbol recorded at the cell $\mathbf{c}$ will be $(\sharp, \sharp)$ if $\mathbf{c} \notin p$, $(\sharp, \mathbf{c_2} - \mathbf{c_1})$ if $\mathbf{c} = \mathbf{c_1}$, $(\mathbf{c_n} - \mathbf{c_{n-1}}, \sharp)$ if $\mathbf{c} = \mathbf{c_n}$ or $(\mathbf{c_i} - \mathbf{c_{i-1}}, \mathbf{c_{i+1}} - \mathbf{c_i})$ otherwise. It indicates how the path enters and exits the cell $\mathbf{c}$.

Definition 2.5. To specify language recognition by CA, we need to distinguish three subsets of the set of states S: Σ the input alphabet, S_{accept} the set of accepting states and S_{reject} the set of rejecting states. We also identify the cell $\mathbf{0} = (0, 0)$ as the output cell which determines the acceptance. And we have to precise the input mode. For a CA with a path, the input word is fed in parallel along this path: the i-th symbol of the input word is gotten on the i-th position of the path. Precisely, given the input word $w = w_1 \cdots w_n \in \Sigma^*$ and the path $p = (\mathbf{c_1}, \cdots, \mathbf{c_n})$, the cells are set up at initial time in these states:

$$\langle \mathbf{c}, 0 \rangle = \begin{cases} ((\sharp, \sharp), \varepsilon) & \text{if } \mathbf{c} \notin p \\ ((\sharp, \mathbf{c_2} - \mathbf{c_1}), w_1) & \text{if } \mathbf{c} = \mathbf{c_1} \\ ((\mathbf{c_n} - \mathbf{c_{n-1}}, \sharp), w_n) & \text{if } \mathbf{c} = \mathbf{c_n} \\ ((\mathbf{c_i} - \mathbf{c_{i-1}}, \mathbf{c_{i+1}} - \mathbf{c_i}), w_i) & \text{otherwise} \end{cases}$$

We say that L is recognized by a CA $\mathcal{A}$ with a path p if on input $w \in \Sigma^*$, the output cell enters an accepting state if $w \in L$ or a rejecting state if $w \notin L$ at some time t_0; and for all time $t < t_0$, the output cell is neither in an accepting or rejecting state.

Definition 2.6. The diameter of the path relatively the von Neumann neighborhood and the cell $\mathbf{0}$ is the minimal radius of the von Neumann ball of center $\mathbf{0}$ which encompasses the path. L is recognized in real time by $\mathcal{A}$ with p if the output cell $\mathbf{0}$ enters an accepting or rejecting state at step diameter$(p) - 1$. That is the minimal time that the output cell $\mathbf{0}$ knows the whole path.

3. The algorithm

While dealing with two-dimensional CA recognizer, specific problems arise when the cells do not know the position of the output cell. To avoid the problems, we suppose that every cell knows the relative distances of its neighbors from each other relatively to the output cell. By instance, with von Neumann neighborhood, a cell in the positive quadrant knows that its closest neighbors relatively to the output cell $\mathbf{0}$ are its north and west ones, and its farthest neighbors are its south and east ones.

Proposition 3.1. *Providing each cell knows the position of the output cell, any regular language is recognized in real time modulo a constant by a CA equipped with a path, whatever the path may be.*

3.1. The outline

For the sake of simplicity, we may suppose that the path is in the positive quadrant and so that each cell involved in the computation knows that the position of the output cell is in the direction of the northwest. The approach is based on the strategy used by Levialdi [4, 5]

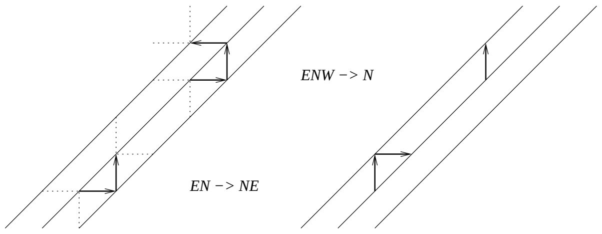


Figure 1: The rewriting rules

which applies local transformations in parallel to shrink object in such a way the diameter of the object decreases of one at each step. Here the CA will shrink the path in applying local rewriting rules to its sequence of moves: $\mathbf{sen} \to \mathbf{e}$, $\mathbf{swn} \to \mathbf{w}$, $\mathbf{enw} \to \mathbf{n}$, $\mathbf{esw} \to \mathbf{s}$, $\mathbf{sw} \to \mathbf{ws}$, $\mathbf{en} \to \mathbf{ne}$; and for the extremities of the path, $\sharp\mathbf{n} \to \sharp$, $\sharp\mathbf{w} \to \sharp$, $\mathbf{s}\sharp \to \sharp$, $\mathbf{e}\sharp \to \sharp$. See Figure 1.

In the same time, the CA will record, above the path, all possible transitions induced locally by the finite automata. Initially, that is, for the cell $\mathbf{c_i}$ which gets the input symbol w_i, the set of transitions $\{(q, \delta(q, w_i)) : q \in Q\}$. Remark that, given the computation which starts in initial state q_{init}, passes through the states $q_1, \cdots, q_{n-1}$ and ends in state f of the input word w, the tuples $(q_{init}, q_1), (q_1, q_2), \cdots, (q_{n-2}, q_{n-1}), (q_{n-1}, f)$ will be recorded on the cells $\mathbf{c_1}, \mathbf{c_2}, \cdots, \mathbf{c_{n-1}}, \mathbf{c_n}$ respectively. Together with the shrinkage of the path, the aim will be to update and reduce this sequence of transitions until it will be shorten to the transition (q_{init}, f). Then, according f is or not an accepting state, the result will be transmitted to the output cell.

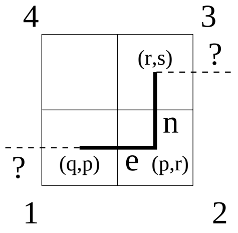


Figure 2: Carrying the rewriting rule $\mathbf{en} \to \mathbf{ne}$

Let us have a look on the rewriting rule $\mathbf{en} \to \mathbf{ne}$ performed at some time t. See Figure 2. Consider $\mathbf{c}$, $\mathbf{c}+\mathbf{e}$, $\mathbf{c}+\mathbf{s}$, $\mathbf{c}+\mathbf{e}+\mathbf{s}$ the four cells involved in the process. At time $t-1$, the path is coded by the three tuples $(?, \mathbf{e})$, $(\mathbf{e}, \mathbf{n})$ and $(\mathbf{n}, ?)$ recorded respectively on the cells $\mathbf{c}+\mathbf{e}$, $\mathbf{c}+\mathbf{e}+\mathbf{s}$ and $\mathbf{c}+\mathbf{s}$. In addition, the cells record sets of transitions. For example, (q,p) belongs to the set of $\mathbf{c}+\mathbf{e}$, (p,r) to $\mathbf{c}+\mathbf{e}+\mathbf{s}$ and (r,s) to $\mathbf{c}+\mathbf{s}$. Now at step t, the four cells have all the required information to perform the rewriting rule $\mathbf{en} \to \mathbf{ne}$, in particular the cell $\mathbf{c}$ can compute the tuple $(\mathbf{n}, \mathbf{e})$ but it needs one more step to compute the transition (p,r). To avoid this slowdown, we will introduce the two following moves: the eastnorth move $\widetilde{\mathbf{en}} = (-1,1)$ and the southwest $\widetilde{\mathbf{sw}} = (1,-1)$. And every two consecutive moves $\mathbf{e}$ and $\mathbf{n}$ (respectively $\mathbf{s}$ and $\mathbf{w}$) will be coded by only one move $\widetilde{\mathbf{en}}$ (respectively $\widetilde{\mathbf{sw}}$). By the way, the rewriting rules will be modified in this following manner: $\mathbf{n}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{n} \to \mathbf{n}\,\mathbf{n}\,\widetilde{\mathbf{en}}^{\mathbf{k}}$, $\mathbf{e}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{n} \to \widetilde{\mathbf{en}}^{\mathbf{k+1}}$, $\mathbf{s}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{n} \to \widetilde{\mathbf{en}}^{\mathbf{k}}$, $\mathbf{n}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{w} \to \mathbf{n}\,\mathbf{n}\,\widetilde{\mathbf{en}}^{\mathbf{k-1}}$, $\mathbf{e}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{w} \to \widetilde{\mathbf{en}}^{\mathbf{k}}$, $\mathbf{s}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{w} \to \widetilde{\mathbf{en}}^{\mathbf{k-1}}$, $\mathbf{n}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{e} \to \mathbf{n}\,\mathbf{n}\,\widetilde{\mathbf{en}}^{\mathbf{k-1}}\mathbf{e}\,\mathbf{e}$, $\mathbf{e}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{e} \to \widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{e}\,\mathbf{e}$, $\mathbf{s}\,\widetilde{\mathbf{en}}^{\mathbf{k}}\,\mathbf{e} \to \widetilde{\mathbf{en}}^{\mathbf{k-1}}\mathbf{e}\,\mathbf{e}$ and with all the dual rewriting rules ($\mathbf{s}\,\widetilde{\mathbf{sw}}^{\mathbf{k}}\,\mathbf{s} \to \widetilde{\mathbf{sw}}^{\mathbf{k}}\,\mathbf{s}\,\mathbf{s}$, ...) obtained in inverting the direction of the path. Now, at the initialization, we will lose one step to code the path in the right way. But, after, the sequences of transitions will be updated at the same rhythm than the shrinkage of the path.

3.2. The description of the CA

Let be given a FA $\mathcal{F} = (\Sigma, Q, \delta, q_{init}, Q_{\text{accept}})$. The purpose of this section is to describe a CA $\mathcal{A} = (P, S, V, \delta_{\mathcal{A}})$ equipped with a path which recognizes the language $L(\mathcal{F})$.

3.2.1. *The set of states.* First let us define the set of states. It is $S = \Sigma \cup R \times \mathcal{P}(Q^2)$ where $R = \{\mathbf{n}, \mathbf{s}, \mathbf{e}, \mathbf{w}, \widetilde{\mathbf{en}}, \widetilde{\mathbf{sw}}, \sharp\} \times \{\mathbf{n}, \mathbf{s}, \mathbf{e}, \mathbf{w}, \widetilde{\mathbf{en}}, \widetilde{\mathbf{sw}}, \sharp\} \setminus \{(\mathbf{n}, \mathbf{s}), (\mathbf{s}, \mathbf{n}), (\mathbf{e}, \mathbf{w}), (\mathbf{w}, \mathbf{e}), (\mathbf{e}, \mathbf{n}), (\mathbf{s}, \mathbf{w}), (\widetilde{\mathbf{en}}, \mathbf{s}), (\mathbf{n}, \widetilde{\mathbf{sw}}), (\mathbf{w}, \widetilde{\mathbf{en}}), (\widetilde{\mathbf{sw}}, \mathbf{e}), (\widetilde{\mathbf{en}}, \widetilde{\mathbf{sw}}), (\widetilde{\mathbf{sw}}, \widetilde{\mathbf{en}})\}$. Note that the states s belonging to Σ only occur at initial time 0. For states $s = (s_{\text{dir}}, s_{\text{trans}})$ belonging to $R \times \mathcal{P}(Q^2)$, the first component s_{dir} in R codes how the path enters and exits the cell. Because the initial path is simple as well as its rewritings, some consecutive moves never occur. The second component s_{trans} in $\mathcal{P}(Q^2)$ records the set of transitions.

3.2.2. *The preliminary step.* Second let us describe the first step. At initial time 0, the additional layer codes the initial path $p = (\mathbf{c_1}, \cdots, \mathbf{c_n})$ using the set of symbols $P = \{\mathbf{n}, \mathbf{s}, \mathbf{e}, \mathbf{w}, \sharp\} \times \{\mathbf{n}, \mathbf{s}, \mathbf{e}, \mathbf{w}, \sharp\} \setminus \{(\mathbf{n}, \mathbf{s}), (\mathbf{s}, \mathbf{n}), (\mathbf{e}, \mathbf{w}), (\mathbf{w}, \mathbf{e})\}$ and the input words symbols w_i are gotten on the cells $\mathbf{c_i}$. The aim of this preliminary step is to replace every consecutive moves $\mathbf{e}$ and $\mathbf{n}$ (respectively $\mathbf{s}$ and $\mathbf{w}$) of the path by the unique move $\widetilde{\mathbf{en}}$ (respectively $\widetilde{\mathbf{sw}}$). And moreover to record above this path, the transitions induced locally by the FA $\mathcal{F}$. Actually, in one step, all cells have the required information to do the job:

- If the cell $\mathbf{c}$ is outside of the path or the path enters from the East on the cell $\mathbf{c}$ and exits to the North or the path enters from the South and exits to the West (in other words the symbol on the additional layer is $(\sharp, \sharp)$, $(\mathbf{e}, \mathbf{n})$ or $(\mathbf{s}, \mathbf{w})$) then the cell $\mathbf{c}$ enters the quiescent state: $\langle \mathbf{c}, 1\rangle_{\text{dir}} = (\sharp, \sharp)$ and $\langle \mathbf{c}, 1\rangle_{\text{trans}} = \{\}$.
- In case $\mathbf{c}$ is the starting extremity $\mathbf{c_1}$ of the path, if the symbols on the additional layer at position $\mathbf{c}$ and $\mathbf{c} + \mathbf{e}$ (respectively $\mathbf{c}$ and $\mathbf{c} + \mathbf{s}$) are $(\sharp, \mathbf{e})$ and $(\mathbf{e}, \mathbf{n})$ (respectively $(\sharp, \mathbf{s})$ and $(\mathbf{s}, \mathbf{w})$) then $\langle \mathbf{c}, 1\rangle_{\text{dir}} = (\sharp, \widetilde{\mathbf{en}})$ (respectively $(\sharp, \widetilde{\mathbf{sw}})$) and $\langle \mathbf{c}, 1\rangle_{\text{trans}} = \{(q_{init}, \delta(q_{init}, w_1 w_2))\}$. Otherwise $\langle \mathbf{c}, 1\rangle_{\text{dir}}$ takes as value the symbol on the additional layer and $\langle \mathbf{c}, 1\rangle_{\text{trans}} = \{(q_{init}, \delta(q_{init}, w_1))\}$.
- In case $\mathbf{c}$ is the ending extremity $\mathbf{c_n}$ of the path, if the symbols on the additional layer at position $\mathbf{c}$ and $\mathbf{c} + \mathbf{s}$ (respectively $\mathbf{c}$ and $\mathbf{c} + \mathbf{e}$) are $(\mathbf{n}, \sharp)$ and $(\mathbf{e}, \mathbf{n})$ (respectively $(\mathbf{w}, \sharp)$ and $(\mathbf{s}, \mathbf{w})$) then the first component $\langle \mathbf{c}, 1\rangle_{\text{dir}}$ is $(\widetilde{\mathbf{en}}, \sharp)$ (respectively $(\widetilde{\mathbf{sw}}, \sharp)$) otherwise it takes as value the symbol on the additional layer. The second component $\langle \mathbf{c}, 1\rangle_{\text{trans}}$ is $\{(q, f) \in Q^2 : \delta(q, w_n) = f\}$.
- For the remaining cells $\mathbf{c_i}$ on the path, the first component $\langle \mathbf{c_i}, 1\rangle_{\text{dir}}$ is defined in the same way as for the extremities. The second component $\langle \mathbf{c_i}, 1\rangle_{\text{trans}}$ is $\{(q, \delta(q, w_i w_{i+1})) : q \in Q\}$ if the symbols on the additional layer at position $\mathbf{c}$ and $\mathbf{c} + \mathbf{e}$ (respectively $\mathbf{c}$ and $\mathbf{c} + \mathbf{s}$) are $(\sharp, \mathbf{e})$ and $(\mathbf{e}, \mathbf{n})$ (respectively $(\sharp, \mathbf{s})$ and $(\mathbf{s}, \mathbf{w})$) and $\{(q, \delta(q, w_i)) : q \in Q\}$ otherwise.

3.2.3. *The transition function relative to the first component.* Third let us describe the transition function relative to the first component. The different situations required by our CA are depicted in Figure 3. Observe that the north and the west neighbors have no impact in the shrinking process as the output cell is situated in the northwest. Moreover, no direction is given as the rewriting process does not depend on the orientation of the path. For example, the first rule depicts the case where $\langle \mathbf{c}, t\rangle_{\text{dir}} = (\sharp, \sharp)$, $\langle \mathbf{c} + \mathbf{s}, t\rangle_{\text{dir}} = (\mathbf{n}, \widetilde{\mathbf{en}})$ and $\langle \mathbf{c} + \mathbf{e}, t\rangle_{\text{dir}} = (\widetilde{\mathbf{en}}, \mathbf{e})$ as well as the case where $\langle \mathbf{c}, t\rangle_{\text{dir}} = (\sharp, \sharp)$, $\langle \mathbf{c} + \mathbf{e}, t\rangle_{\text{dir}} = (\mathbf{w}, \widetilde{\mathbf{sw}})$ and $\langle \mathbf{c} + \mathbf{s}, t\rangle_{\text{dir}} = (\widetilde{\mathbf{sw}}, \mathbf{s})$. All other possible combinations lead to the quiescent tuple $(\sharp, \sharp)$. Actually some combinations never occur because the initial path is simple as well as its rewritings. We also omit combinations where the extremities are involved. They can be performed in a similar way.

3.2.4. *The transition function relative to the second component.* Finally let us define the updating of the second component of the state. We have to specify how locally the sequence of transitions is either shortened, expanded or shifted according the rewriting of the path. As a state of a FA has at most one successor but possibly several predecessors, the transitions are not symmetrical and their updating depends on the orientation of the path. Below we consider only northeast orientation.

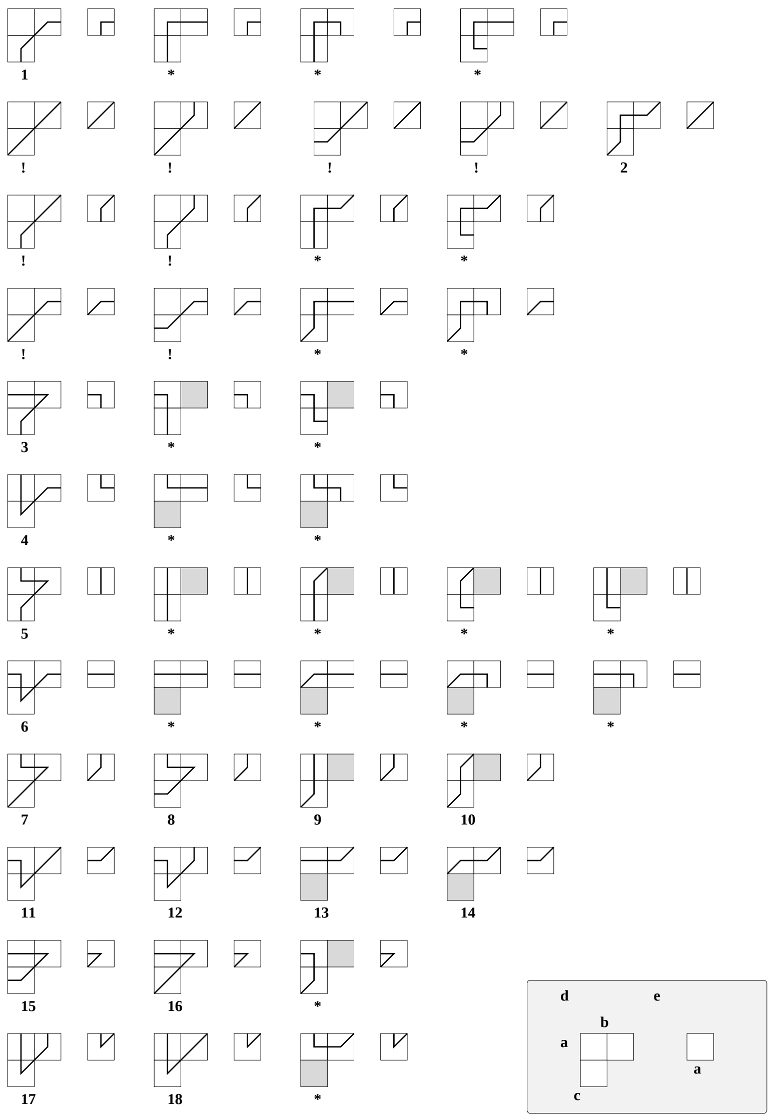


Figure 3: The rules relative to the first component

- For the rules marked with a ’$*$’ in Figure 3, there is no change: $\langle \mathbf{c}, t+1\rangle_{\text{trans}} = \langle \mathbf{c}, t\rangle_{\text{trans}}$.
- For the only rule numbered 1, it is an expansion: $\langle \mathbf{c}, t+1\rangle_{\text{trans}} = \{(q,q) \in Q^2 : \exists p, r$ such that $(p,q) \in \langle \mathbf{c}+\mathbf{s}, t\rangle_{\text{trans}}$ and $(q,r) \in \langle \mathbf{c}+\mathbf{e}, t\rangle_{\text{trans}}\}$.
- For the rules marked with a ’Ⓢ’, it is a shift: $\langle \mathbf{c}, t+1\rangle_{\text{trans}} = \langle \mathbf{c}+\mathbf{e}, t\rangle_{\text{trans}}$.
- For the remaining rules, it is a shrinkage. Precisely $\langle \mathbf{c}, t+1\rangle_{\text{trans}} = \{(p,q) \in Q^2 : \exists r$ such that $(p,r) \in \langle \mathbf{a}, t\rangle_{\text{trans}}$ and $(r,q) \in \langle \mathbf{b}, t\rangle_{\text{trans}}\}$ with $\mathbf{a} = \mathbf{c}+\mathbf{e}$ and $\mathbf{b} = \mathbf{c}$ for the rules $2, 3, 5, 7, 8, 15$, with $\mathbf{a} = \mathbf{c}$ and $\mathbf{b} = \mathbf{c}+\mathbf{s}$ for the rules $4, 6, 11, 12, 17,$

with $\mathbf{a} = \mathbf{c} + \mathbf{s}$ and $\mathbf{b} = \mathbf{c}$ for the rules $9, 10$, and with $\mathbf{a} = \mathbf{c}$ and $\mathbf{b} = \mathbf{c} + \mathbf{e}$ for the rules $13, 14$. And $\langle \mathbf{c}, t+1 \rangle_{\text{trans}} = \{(p, q) \in Q^2 : \exists r, s$ such that $(p, r) \in \langle \mathbf{a}, t \rangle_{\text{trans}}$ and $(r, s) \in \langle \mathbf{b}, t \rangle_{\text{trans}}$ and $(s, q) \in \langle \mathbf{d}, t \rangle_{\text{trans}}\}$ with $\mathbf{a} = \mathbf{c} + \mathbf{s}$, $\mathbf{b} = \mathbf{c} + \mathbf{e}$, $\mathbf{d} = \mathbf{c}$ for the rule 16 and $\mathbf{a} = \mathbf{c}$, $\mathbf{b} = \mathbf{c} + \mathbf{s}$, $\mathbf{d} = \mathbf{c} + \mathbf{e}$ for the rule 18.

An example is given in Figure 4. The tuples of letters along the path represent the computation of the finite automaton. Note that each tuple is just an element of the set of tuples recorded by the second component.

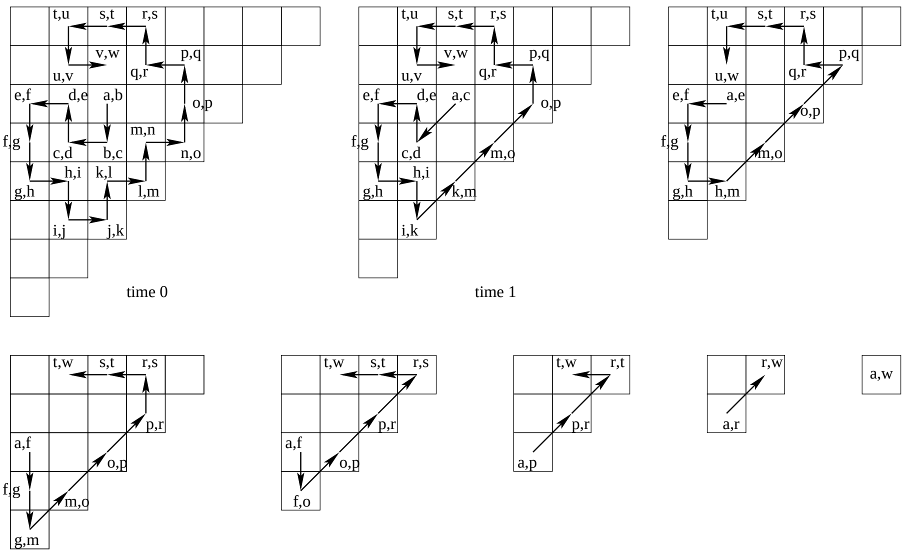


Figure 4: The shrinking process

3.2.5. *Correctness of the algorithm.* According the transition function relative to the first component given in Figure 3, the algorithm rewrites a connected path into a connected path. Moreover, we observe that, if at step t the path does not pass through the south and east borders of the cells $\mathbf{c} + \mathbf{s}$ and $\mathbf{c} + \mathbf{e}$, then at step $t + 1$ the path does not pass through the south and east borders of the cell $\mathbf{c}$. Hence, in one step, the algorithm turns a path of diameter d into a path of diameter $d - 1$. Finally, remark that the connectivity between the initial state and the final state of the computation of the FA, is preserved by the updating of the second component.

4. Conclusion

The algorithm presented in this paper is based on the strong assumption that each cell knows the relative positions of its neighbors from the output cell. At first glance, it seems unlikely to get rid of this hypothesis. Actually, this problem of orientation is a rather general question concerning CA recognizers in dimension 2 and deserves to be clarified.
We have also assumed that the paths are simple. Clearly the algorithm may be adapted for any path which can go through the same cell more than once but it remains the essential

condition that the path does not cross itself. What happens when we authorize constant bounded crossings of the paths? The algorithm should be modified to avoid unbounded collisions: it will achieve as many rewriting rules as possible (as allowed by the state capacity of the CA) and it will delay the other ones for the next steps. A question is to what extent is the slowdown of the global process linked to these local delays.
Another simplification made was to deal with von Neumann neighborhood instead of Moore neighborhood as it was considered in [2]. So we may wonder whether this shrinking algorithm can be extended to Moore neighborhood.

References

[1] K. Culik. Variations of the firing squad problem and applications. *Information Processing Letters*, 30(3):153–157, February 1989.

[2] M. Delorme and J. Mazoyer. Reconnaissance parallèle des langages rationnels sur automates cellulaires plans. *Theoretical Computer Science*, 281(1–2):251–289, May 2002.

[3] P. C. Fischer. Generation of primes by one-dimensional real-time iterative array. *Journal of the ACM*, 12:388–394, 1965.

[4] S. Levialdi. On shrinking binary picture patterns. *Communications of the ACM*, 15(1):7–10, 1972 .

[5] H. Umeo and G. Mauri. A duality theorem for two connectivity-preserving parallel shrinking transformations. *Future Generation Computer Systems*,18 (7):931–937, August 2002.

Journées Automates Cellulaires 2008 (Uzès), pp. 182-194

AMALGAMATION OF CELLULAR AUTOMATA

GUILLAUME THEYSSIER

E-mail address: guillaume.theyssier@univ-savoie.fr

LAMA (UMR 5127), Campus Scientifique, 73376 Le Bourget-du-lac cedex FRANCE
URL: http://www.lama.univ-savoie.fr/~theyssier/

Abstract. In this paper, we study amalgamations of cellular automata (CA), i.e. ways of combining two CA by disjoint union. We show that for several families of CA obtained by simple amalgamation operations (including the well-known families of majority and minority CA), the density of a large class of properties follows a zero-one law. Besides, we establish that intrinsic universality in those families is always non-trivial and undecidable for some of them. Therefore we obtain various syntactical means to produce CA which are almost all intrinsically universal. These results extend properties already obtained for captive cellular automata. We additionally prove that there exists some reversible captive CA which are (intrinsically) universal for reversible CA.

1. Introduction

Cellular automata (CA) are discrete dynamical systems made of an infinite lattice of cells evolving synchronously and uniformly according to a common local rule. The model of cellular automata offers a minimal formal setting to tackle the broad questioning from the field of complex systems: how repetitions of a simple local rule can lead to complex global behaviour?

It is well known through numerous undecidability results [8, 9] that determining global behaviour of CA from their local rule is very challenging. However, the inverse problem of constructing a local rule to achieve a given global behaviour has received a lot of attention in the literature with some success. Hence, one of the most celebrated result of CA theory is the existence of small universal CA (see [2, 3, 15] for smallest known examples in various settings). Historically, the notion of universality used for CA was a more or less formalised adaptation from classical Turing-universality. Later [1, 10, 6, 13], a stronger notion called *intrinsic universality* was proposed: a CA is intrinsically universal if it is able to simulate step by step any other CA. This definition relies on a notion of simulation which, in addition to defining intrinsic universality, provides a sharp formal tool (an infinite pre-order) to classify cellular automata.

As said before, the existence of intrinsically universal CA is clearly established and the present paper (in the continuation of previous work of the author [17]) aims at going further by showing that this property, although highly non-trivial, is in fact very common

Key words and phrases: cellular automata, amalgamation, density, zero-one law, universality.

and "malleable". More precisely, we will show that one can define various families of CA, by purely syntactical constraints on local rules, which have simultaneously the following properties:

- almost all CA of the family are intrinsically universal,
- intrinsic universality is undecidable in the family.

To achieve this, we study various notions of *amalgamation* of CA. By amalgamation, we mean an operation which combines two CA by disjoint union. More precisely, an amalgamation operation consists in defining the behaviour of the local rule for transitions involving states from both CA (see definition 2.4 below). These operations are interesting on their own and some were already specifically studied from different points of view [4, 5]. We give several natural examples of amalgamation operations in section 3 and consider associated families of CA (obtained as the cloture by amalgamation of a finite set of CA). We then show in section 4 that they all satisfy a zero-one law for a large class of properties (including intrinsic universality): a property has either probability 0 or probability 1 in the family. Finally, in section 5, we focus on the class of intrinsically universal CA and show that its intersection with the families above is complex (notably undecidable) although probabilistically trivial. Besides, we prove the existence of reversible captive CA which are (intrinsically) universal for reversible CA (i.e. able to simulate any reversible CA). This last result gives another indication of the "malleability" of the notion of (intrinsic) universality.

2. Definitions

For clarity of exposition and although some of the following results extends to any dimension, we assume throughout the paper that dimension is 1. Moreover, we consider only centered connected neighbourhoods. In this setting, a CA is triple $F = (Q_F, r, \delta_F)$ where Q_F is a finite set of *states*, r (the neighbourhood's radius) is a positive integer δ_F is a map from Q_F^{2r+1} to Q_F. Configurations are maps from $\mathbb{Z}$ to Q_F. The local transition function δ_F induces a global evolution rule on configurations denoted Q_F and defined as follows: $\forall c \in Q_F^{\mathbb{Z}}$, $\forall i \in \mathbb{Z}$, $F(c)(i) = \delta_F\big(c(i-r), c(i-r+1), \ldots, c(i+r)\big)$.

Let F be any CA with state set Q_F. A subset $Q \subseteq Q_F$ is F-*stable* if $F\big(Q^{\mathbb{Z}}\big) \subseteq Q^{\mathbb{Z}}$. Then F induces a cellular automaton on $Q^{\mathbb{Z}}$, denoted by $F_{|Q}$. G with state set Q_G is a *sub-automaton* of F, denoted by $G \sqsubseteq F$, if there is a F-stable subset Q such that G is isomorphic to $F_{|Q}$: formally, there is a one-to-one map $\iota : Q_G \to Q$ such that

$$\iota \circ G(x_{-r}, \ldots, x_r) = F_{|Q}(\iota(x_{-r}), \ldots, \iota(x_r))$$

for any $x_{-r}, \ldots, x_r \in Q_G$.

In the sequel, we denote by $F \equiv G$ the fact that F and G are isomorphic (both $F \sqsubseteq G$ and $G \sqsubseteq F$). Moreover, we say that a state q is quiescent for F if the set $\{q\}$ is F-stable.

The relation $\sqsubseteq$ provides a local comparison relation on CA which is very restrictive. We now define a pre-order relation generalizing $\sqsubseteq$ by allowing some rescaling operations in the CA to be compared. Our formalisation below follows[1] that of [14]. Rescaling transformations considered are very simple: they allow grouping several cells in one block and

[1]To be precise, we don't use the shift parameter present in [14] in order to simplify notations. However all our proofs remains correct when using this parameter in rescaling transformations.

running several steps at a time. Formally, for any finite set A and any $m \in \mathbb{N}$ ($m \neq 0$), let $b_m : A^{\mathbb{Z}} \to (A^m)^{\mathbb{Z}}$ be the map such that

$$\forall c \in A^{\mathbb{Z}}, \forall z \in \mathbb{Z} : \big(b_m(c)\big)(z) = \big(c(mz), c(mz+1), \ldots, c(m(z+1)-1)\big).$$

We denote by $F^{<m,t>}$ the CA $b_m \circ F^t \circ b_m^{-1}$. Once rescaling transformation are defined, the simulation relation is simply the relation $\sqsubseteq$ up to rescaling.

Definition 2.1 (simulation). *F simulates G*, denote by $G \preceq F$, if there are parameters m, t and m', t' such that $G^{<m,t>} \sqsubseteq F^{<m',t'>}$.

From a dynamical systems point of view, if $G \preceq F$ then there exists a set of configurations Σ such that the dynamical system $\big(G^t, Q_G^{\mathbb{Z}}\big)$ is isomorphic to $\big(F^{t'}, \Sigma\big)$. Using the pre-order $\preceq$ as a tool to measure complexity of CA, we will particularly focus on properties which are *increasing* for $\preceq$. A property $\mathcal{P}$ is increasing if whenever $F \preceq G$ then $F \in \mathcal{P} \Rightarrow G \in \mathcal{P}$. A property is decreasing if its complement is increasing.

As evoked in the introduction, the notion of universality used throughout this paper is defined as the global maximum class of $\preceq$.

Definition 2.2 (universality).
- F is *universal* if for any G we have $G \preceq F$.
- F is *reversible-universal* if it is reversible and for any reversible G we have $G \preceq F$.

We denote by $\mathcal{U}$ the set of universal CA.

Throughout this paper we consider several families $\mathcal{F}$ of CA and we are interested in the typical properties of elements of $\mathcal{F}$ when their state set gets larger and larger. This is formalised by the notion of density, that is, the limit probability (of some property) for cellular automata with increasing state set but fixed neighbourhood. Unless it is specified in the context, the neighbourhood radius r is arbitrary and the arity of local rules is denoted by k with $k = 2r + 1$.

Any alphabet considered in this paper is a subset of $\mathbb{N}$. We denote by $\mathtt{CA}$ the set of CA and by $\mathtt{CA}_n$ the set of CA on state set $\{0, \ldots, n-1\}$. Given a CA F we denote by $|F|$ the cardinal of its state set. Moreover, we fix once for all a collection of bijections between state sets of same size (which are always subsets of $\mathbb{N}$): we choose the only bijection which is increasing according to the natural order of $\mathbb{N}$. Thus we get a standard renaming function $\mathtt{std}()$ such that for any $F \in \mathtt{CA}$ and any set X with $|X| = |F|$ we have $\mathtt{std}_X(F) \equiv F$ and the state set of $\mathtt{std}_X(F)$ is X.

Definition 2.3 (density).
- Given a family $\mathcal{F}$ and a property $\mathcal{P}$ (both are sets of CA), we define $d_{\mathcal{F}}(\mathcal{P})$, the *density* of $\mathcal{P}$ in $\mathcal{F}$, as the following limit (if it exists):
$$\lim_{\substack{n \in I_{\mathcal{F}} \\ n \to \infty}} \frac{|\mathcal{P} \cap \mathcal{F}_n|}{|\mathcal{F}_n|},$$
where $\mathcal{F}_n = \mathcal{F} \cap \mathtt{CA}_n$ and $I_{\mathcal{F}} = \{n : \mathcal{F} \cap \mathtt{CA}_n \neq \emptyset\}$.
- In the sequel, when $\mathcal{F} = \mathtt{CA}$, $d_{\mathcal{F}}(\mathcal{P})$ is abbreviated to $d(\mathcal{P})$.

The present paper is devoted to families of CA obtained by amalgamation. Intuitively, an amalgamation consists in making the (disjoint) union of two CA and completing the transition table in some way (i.e. choosing a value for each transition involving states from both CA). Precisely, given two CA of size n and p, there are $(n+p)^k - n^k - p^k$ transitions to fix in order to completely define an amalgamation of them. An amalgamation operation is a description of allowed completion for any pair of CA.

Definition 2.4 (amalgamation)**.** An *amalgamation operation* Γ is function from pair of CA to sets of CA verifying for all $F, G \in \texttt{CA}$:

(1) $\Gamma(F, G) \neq \emptyset$,
(2) $|\Gamma(F, G)|$ is a function of $|F|$ and $|G|$ only,
(3) if $H \in \Gamma(F, G)$ then
  (a) $H \in \texttt{CA}_{p+q}$,
  (b) $H_{|Q} = \texttt{std}_Q(F)$ and
  (c) $H_{|Q'} = \texttt{std}_{Q'}(G)$
  where $p = |F|$, $q = |G|$, $Q = \{0, \ldots, p-1\}$ and $Q' = \{p, \ldots, p+q-1\}$.

Γ is said *associative* if for any F, G, H it verifies:

$$\Gamma\big(F, \Gamma(G, H)\big) = \Gamma\big(\Gamma(F, G), H\big),$$

where the notation Γ is naturally extended to sets of CA.

Given an amalgamation operation Γ, and a finite set of CA $\mathbb{G}$, called *generators*, we consider the set $\mathcal{F}_{\mathbb{G},\Gamma}$ (or simply $\mathcal{F}$ when the context is clear) which is the smallest set containing $\mathbb{G}$ and closed by Γ.

For any $F_1, \ldots, F_n \in \mathbb{G}$, we denote by $\Gamma(F_1, \ldots, F_n)$ the set

$$\Gamma\left(F_1, \Gamma(F_2, \ldots, \Gamma(F_{n-1}, F_n) \cdots)\right).$$

If Γ is associative, any $F \in \mathcal{F}$ belongs to some $\Gamma(F_1, \ldots, F_n)$ for a convenient choice of $F_1, \ldots, F_n$. Moreover, if $F \in \Gamma(G, H) \cap \Gamma(G', H')$ then we have necessarily either $G \sqsubseteq G'$ or $G' \sqsubseteq G$. Therefore, if we suppose that there is no $G, G' \in \mathbb{G}$ with $G \sqsubseteq G'$, then, for any $F \in \mathcal{F}$, there is a unique list of generators $F_1, \ldots, F_n \in \mathbb{G}$ such that $F \in \Gamma(F_1, \ldots, F_n)$ (straightforward by induction on the size of F).

In the sequel, any family $\mathcal{F}_{\mathbb{G},\Gamma}$ with such properties will be called an *unambiguous amalgamation family.*

3. Examples

We will establish some properties shared by all unambiguous and associative amalgamation families in section 4. The present section gives several examples of amalgamation families interesting in their own to illustrate the previous definitions. Before giving examples of amalgamation operations and considering the associated families, we will define 3 natural families of CA already considered in the literature which turn out to be amalgamation families. First, captive CA (introduced in [16]) are automata where the transition function is constrained to always choose a state already present in the neighbourhood.

Definition 3.1. A *captive* CA of arity k is a CA where each transition verifies the captivity constraint: $\delta_F(x_1, \ldots, x_k) \in \{x_1, \ldots, x_k\}$. The set of such CA is denoted by $\mathcal{K}$.

A cellular automaton with 2 states known has the majority vote CA has received a lot of attention in the literature (see for instance [11]). Its transition rule simply consists in choosing the state which has the greatest number of occurrences in the neighbourhood. We will consider generalisations to any number of states of the majority vote CA and its symmetric, the minority vote CA.

Definition 3.2.
- A *majority* CA of arity k is a CA where each transition verifies the majority constraint: $\delta_F(x_1, \ldots, x_k) \in \big\{x_i : \forall j, c(i) \geq c(j)\big\}$, where $c(i) = |\{j : x_j = x_i\}|$. The set of majority CA is denoted by $\mathcal{MAJ}$.

- The set $\mathcal{MIN}$ of *minority* CA is defined analogously, the condition on each transition becoming: $\delta_F(x_1,\dots,x_k) \in \bigl\{x_i : \forall j, c(i) \leq c(j)\bigr\}$.

We now proceed the opposite way and define natural amalgamation operations in order to consider associated amalgamation families in the sequel.

First, we define the general amalgamation operation Γ_g as the one where all possible completion of transition tables are allowed. Γ_g is obviously non-associative since there are some $G \in \Gamma_g\bigl(F_1, \Gamma_g(F_2, F_3)\bigr)$ having transitions involving only states corresponding (through `std()`) to F_1 and F_2 which leads to some state corresponding to F_3: this is impossible in $\Gamma_g\bigl(\Gamma_g(F_1, F_2), F_3\bigr)$.

Since associativity plays an important role in determining the density of properties in amalgamation families, we define the amalgamation operation Γ_a obtained by adding the following restriction: any completion of the transition table must ensure when possible that the union of any pair of stable subsets is itself a stable subset.

Definition 3.3. Γ_a is defined as follows: for any $F \in \texttt{CA}_p$ and $G \in \texttt{CA}_q$, $\Gamma_a(F, G)$ is the set of all automata $G \in \texttt{CA}_{p+q}$ such that if $Q = \{0, \dots, p-1\}$ and $Q' = \{p, \dots, p+q-1\}$:

(1) $H_{|Q} = \texttt{std}_Q(F)$ and $H_{|Q'} = \texttt{std}_{Q'}(G)$,
(2) for any $L \subseteq Q$ and $R \subseteq Q'$, if both L and R are H-stable then $L \cup R$ is also H-stable.

Proposition 3.4. *Γ_a is associative.*

Proof. Let $F_1, F_2, F_3 \in \texttt{CA}$ and consider any $G \in \Gamma_a(F_1, \Gamma_a(F_2, F_3))$. Denote by Q_1, Q_2 and Q_3 the state sets corresponding to F_1, F_2 and F_3 (respectively) in G. First it is straigthforward to check that $G_{|Q_1 \cup Q_2} \in \Gamma_a(F_1, F_2)$.

Now consider any $L \subseteq Q_1 \cup Q_2$ and $R \subseteq Q_3$ which are both G-stable. Let $L_1 = L \cap Q_1$ and $L_2 = L \cap Q_2$. $L_2 \cup R$ is G-stable because $G_{|Q_2 \cup Q_3} \in \Gamma_a(F_2, F_3)$. Therefore $L \cup R$ is G-stable because it is the union of stable sets $L_1 \subseteq Q_1$ and $L_2 \cup R \subseteq Q_2 \cup Q_3$ and $G \in \Gamma_a(F_1, \Gamma_a(F_2, F_3))$. Hence $G \in \Gamma_a(\Gamma_a(F_1, F_2), F_3)$. ∎

One can notice that the condition on stable subsets satisfied by Γ_a is a necessary condition for associativity (remark on Γ_g above). Thus, Γ_a is the largest associative amalgamation operation.

The 3 families $\mathcal{K}, \mathcal{MIN}, \mathcal{MAJ}$ rely on a constraint applying to each transition individually. It is thus natural to consider for each one the amalgamation operation where any completion in the transition table fulfils the constraint.

Definition 3.5.
- $\Gamma_{\mathcal{K}}$ is the amalgamation operation defined as follows: for any $F \in \texttt{CA}_p$ and $G \in \texttt{CA}_q$, $\Gamma_{\mathcal{K}}(F, G)$ is the set of all automata $G \in \Gamma_a(F, G)$ such that any transition involving states from both $\{0, \dots, p-1\}$ and $\{p, \dots, p+q-1\}$ satisfy the captivity constraint.
- The amalgamation operations $\Gamma_{\mathcal{MIN}}$ and $\Gamma_{\mathcal{MAJ}}$ are defined analogously using the minority and majority constraint respectively.

Proposition 3.6. *Let $\mathbb{G}_0$ be the set containing only the trivial CA with a single state. Each of the sets $\mathcal{K}$, $\mathcal{MIN}$ and $\mathcal{MAJ}$ forms an unambiguous amalgamation family associated to $\mathbb{G}_0$ and $\Gamma_{\mathcal{K}}$, $\Gamma_{\mathcal{MIN}}$ or $\Gamma_{\mathcal{MAJ}}$ (respectively).*

Proof. We consider the family $\mathcal{K}$ (proofs for $\mathcal{MIN}$ and $\mathcal{MAJ}$ are similar). First, it is straightforward to check that $\Gamma_{\mathcal{K}}$ is associative. Moreover, since any state q of any captive

CA F is quiescent, it follows that $F \in \Gamma_{\mathcal{K}}(\underbrace{\mathbb{G}_0, \ldots, \mathbb{G}_0}_{|F|})$. Conversely, any F in $\Gamma_{\mathcal{K}}(\mathbb{G}_0, \ldots, \mathbb{G}_0)$ is a captive CA since it has only quiescent states and all other transitions fulfils the captivity constraint by definition of $\Gamma_{\mathcal{K}}$. ■

4. Density of Properties

In this section, we are interested in typical properties of CA from a given amalgamation family. First, one can establish that any amalgamation family is negligible in the set CA. Therefore, the typical behaviour of CA from some amalgamation family is *a priori* not related to the typical behaviour of CA in general. In the next section, we will establish that typical behaviour of several amalgamation families is interesting (theorem 5.4), whereas nothing is known for CA in general.

Proposition 4.1. *If $\mathcal{F}$ is an amalgamation family then $d(\mathcal{F}) = 0$.*

Proof. This is a straightforward corollary of proposition 1 of [17] since by definition any sufficiently large $F \in \mathcal{F}$ possesses a non-trivial sub-automaton. ■

We will now focus on unambiguous amalgamation families and establish the main result of this section which gives a simple sufficient condition for a property to have density 1 in such a family.

Definition 4.2. Given a set of generators $\mathbb{G}$ and an amalgamation operation Γ, we say that a property $\mathcal{P}$ is *malleable* for $\mathbb{G}$ and Γ if the two following conditions hold:

(1) $\mathcal{P}$ is increasing,
(2) there is i such that, for any $F_1, \ldots, F_i \in \mathbb{G}$, $\Gamma(F_1, \ldots, F_i) \cap \mathcal{P} \neq \emptyset$.

It is straightforward to check that for an associative amlgamation operation and a set of generators which is a singleton, the malleable properties are exactly the increasing (non-void) properties. this is the case for $\mathcal{K}$, $\mathcal{MIN}$ and $\mathcal{MAJ}$. Concerning Γ_a and a non-trivial set of generators, we have the following proposition.

Proposition 4.3. *Let $\mathbb{G}$ be any set of generators. Any increasing property $\mathcal{P}$ containing a captive CA is a malleable property for $\mathbb{G}$ and Γ_a.*

Proof. Consider any $F \in \mathcal{P} \cap \mathcal{K}$ and let $\{a_1, \ldots, a_i\}$ denotes its state set. Let $F_1, \ldots, F_i \in \mathbb{G}$. For any $1 \leq j \leq i$ there is a positive integer t_j and a state q_j of F_j such that q_j is a quiescent state of $F_j^{t_j}$ (because the phase space of F_j restricted to uniform configurations necessarily contains a cycle). Now let $m = \operatorname{lcm}(t_j)$ and denote by e_j the state of any CA from $\Gamma_a(F_1, \ldots, F_i)$ corresponding to q_j (for any $1 \leq j \leq i$). Since F is captive, one can choose $H \in \Gamma_a(F_1, \ldots, F_i)$ such that : for any k-tuple $j_1, \ldots, j_k \in \{1, \ldots, i\}$ with at least 2 distinct elements:

$$\delta_F(a_{j_1}, \ldots, a_{j_k}) = a_\alpha \implies \delta_H(e_{j_1}, \ldots, e_{j_k}) = e_\alpha.$$

Hence, although the set $\{e_1, \ldots, e_i\}$ may not be H-stable, the only potential obstruction to this stability comes from the fact that states e_j are not all quiescent for H. In any case, we have:

$$F^m \sqsubseteq H^m$$

and thus $F \preceq H_i$ and $H \in \mathcal{P}$ which concludes the proof. ■

We now establish the central result of this section.

Theorem 4.4. *Let $\mathcal{F} = \mathcal{F}_{\mathbb{G},\Gamma}$ be any unambiguous amalgamation family. Then, for any malleable property $\mathcal{P}$, we have $d_{\mathcal{F}}(\mathcal{P}) = 1$.*

Proof. Let α and β denote the minimum and maximum size (respectively) of elements of $\mathbb{G}$. By unambiguity of $\mathcal{F}$, we can partition $\mathcal{F}_n$ for any $n \in I_{\mathcal{F}}$ according to:

$$\mathcal{F}_n = \bigcup_{\substack{n/\beta \leq p \leq n/\alpha,\\ F_1,\dots,F_p \in \mathbb{G},\\ \sum |F_i| = n}} \Gamma\big(F_1, \dots, F_p\big).$$

Let us fix some p with $n/\beta \leq p \leq n/\alpha$ and some list of generators $F_1, \dots, F_p \in \mathbb{G}$ with $\sum |F_i| = n$. We will show that there exist some $0 \leq \lambda < 1$ depending only on $\mathcal{P}$, $\mathbb{G}$ and Γ such that we have for sufficiently large p:

$$\frac{|\Gamma(F_1, \dots, F_p) \setminus \mathcal{P}|}{|\Gamma(F_1, \dots, F_p)|} \leq \lambda^p.$$

By the above partition, this property implies

$$\frac{|\mathcal{F}_n \setminus \mathcal{P}|}{|\mathcal{F}_n|} \leq \lambda^{n/\beta}$$

which concludes the proof.

By malleability of $\mathcal{P}$ there exists i such that we have $\Gamma(F_{ji+1}, \dots, F_{(j+1)i}) \cap \mathcal{P} \neq \emptyset$ for any $0 \leq j < \lfloor \frac{p}{i} \rfloor$. Moreover, since Γ is associative, one has the following partition

$$\Gamma(F_1, \dots, F_p) = \bigcup_{\substack{G_j \in \Gamma(F_{ji+1},\dots,F_{(j+1)i})\\ \forall j,\ 0 \leq j < \lfloor \frac{p}{i} \rfloor}} \Gamma\big(G_0, \dots, G_{\lfloor \frac{p}{i} \rfloor - 1}, \Gamma(F_{i\lfloor \frac{p}{i} \rfloor + 1}, \dots, F_p)\big).$$

By definition of amalgamation operations, each set of the above partition has the same cardinal. Therefore, if we denote by ϵ the proportion of $\lfloor \frac{p}{i} \rfloor$-tuples $(G_0, \dots, G_{\lfloor \frac{p}{i} \rfloor - 1})$ from the above list such that $G_j \notin \mathcal{P}$ for every j, then we have:

$$\frac{|\Gamma(F_1, \dots, F_p) \setminus \mathcal{P}|}{|\Gamma(F_1, \dots, F_p)|} \leq \epsilon.$$

Finally, let m depending only on $\mathbb{G}$, Γ and $\mathcal{P}$ be defined by

$$m = \max\{|G| : G \in \Gamma\big(\underbrace{\mathbb{G}, \dots, \mathbb{G}}_{i}\big)\}.$$

By choice of i (malleability of $\mathcal{P}$), we have for any sufficiently large p:

$$\epsilon \leq \left(\frac{m-1}{m}\right)^{\lfloor \frac{p}{i} \rfloor}.$$

Thus, choosing $\lambda = \left(\frac{m-1}{m}\right)^{\frac{1}{2i}}$, we have the desired property. ∎

Given a family $\mathcal{F}$ of CA, we say that it has the *zero-one law for monotone properties* if any property $\mathcal{P}$ which is non-trivial in $\mathcal{F}$ (i.e. both $\mathcal{F} \cap \mathcal{P} \neq \emptyset$ and $\mathcal{F} \setminus \mathcal{P} \neq \emptyset$) verifies:

- if $\mathcal{P}$ is increasing then $d_{\mathcal{F}}(\mathcal{P}) = 1$,
- if $\mathcal{P}$ is decreasing then $d_{\mathcal{F}}(\mathcal{P}) = 0$.

We have shown that malleable properties are of density 1 in unambiguous amalgamation families (theorem 4.4), and that some sets of increasing properties are malleable for some unambiguous amalgamation families (proposition 4.3 and remark above). Therefore we have the following corollary.

Corollary 4.5. *Each of the following families has the zero-one law for monotone properties:*

(1) $\mathcal{K}$,
(2) $\mathcal{MIN}$,
(3) $\mathcal{MAJ}$.

Moreover, any unambiguous family associated to Γ_a *verifies the zero-one law for monotone properties which are non-trivial in* $\mathcal{K}$.

5. Universality

In this section, we are interested in the intersection of $\mathcal{U}$ with the different families considered until now. In order to show that these intersection are generally non-empty, we will study the general problem of how to encode CA from a family into CA of another family preserving the intersection with $\mathcal{U}$.

Formally, given two families of CA $\mathcal{F}_1$ and $\mathcal{F}_2$, a computable injective function $\Phi : \mathcal{F}_1 \to \mathcal{F}_2$ is an *encoding* form $\mathcal{F}_1$ into $\mathcal{F}_2$ if for all $F_1 \in \mathcal{F}_1$ we have: $F_1 \preceq \Phi(F_1)$. It is *faithful* if we additionally have $F_1 \in \mathcal{U} \iff \Phi(F_1) \in \mathcal{U}$.

Recall that our convention throughout the paper is to consider CA of fixed neighbourhood. Thus, an encoding send CA of a family to CA of another family with the same neighbourhood. In the sequel we will use the following alternative characterisation of universal CA proved in [12].

Proposition 5.1. $F \in \mathcal{U}$ *if and only if for any* G*, there are parameters* m, t *such that* $G \sqsubseteq F^{<m,t>}$.

We now give encoding results of captive cellular automata into different families.

Proposition 5.2. *There exists an encoding from* $\mathcal{K}$ *to* $\mathcal{MAJ}$ *and an encoding from* $\mathcal{K}$ *to* $\mathcal{MIN}$.

Proof. Let $F \in \mathcal{K}$ and let Q its state set. Denote by H the CA from $\mathcal{MAJ}$ defined as follows[2]:

$$\delta_H(x_1, \dots, x_k) = \max\{x_i \ : \ \forall j, c(i) \geq c(j)\},$$

where $c(i) = |\{j : x_j = x_i\}|$. Now let $Q' = Q \times \{1, \dots, k\}$ and denote by π_1 and π_2 the projection from Q' to Q and from Q' to $\{1, \dots, k\}$ (respectively). We furthermore denote

[2]This particular choice of H is not important to establish the proposition. However we conjecture that this particular choice ensure that the encoding is faithful.

by X the set of words on alphabet $\{1,\ldots,k\}$ of the form $i\cdot(i+1)\cdots k\cdot 1\cdots(i-1)$. We define the CA G (in a computable way) as follows:

$$\delta_G(x_1,\ldots,x_k)=\begin{cases}\big(\delta_F(\pi_1(x_1),\ldots,\pi_1(x_k)),\pi_2(x_{\lfloor\frac{k}{2}\rfloor+1})\big) & \text{if } \pi_2(x_1)\cdots\pi_2(x_k)\in X,\\ \delta_H(x_1,\ldots,x_k) & \text{else.}\end{cases}$$

Since the first case of the above definition implies that there are no two occurrences of a same state in the neighbourhood, and by choice of H, we have $G\in\mathcal{MAJ}$. Moreover, considering the set of configuration whose second component is periodic of period $1\cdot 2\cdots k$, one verifies that definition of G implies $F^{<k,1,0>}\sqsubseteq G^{<k,1,0>}$ and therefore $F\preceq G$.

The construction of an encoding from $\mathcal{K}$ to $\mathcal{MIN}$ is very similar. Indeed, the simulation takes place on a set of configuration where the minority/majority constraints reduce to the captivity constraint. ∎

Concerning unambiguous amalgamation families associated to Γ_a, one can establish a stronger result.

Proposition 5.3. *Let $\mathbb{G}$ be a set generators such that $\mathbb{G}\cap\mathcal{U}=\emptyset$ and let $\mathcal{F}$ be the family associated to $\mathbb{G}$ and Γ_a. There exists a faithful encoding from $\mathcal{K}$ to $\mathcal{F}$.*

Proof. The faithful encoding relies on a modified version of the construction in the proof of proposition 4.3. Let $F\in\mathcal{K}$ with state set $\{a_1,\ldots,a_i\}$ and consider any $F_1,\ldots,F_i\in\mathbb{G}$. Using the same notation let $Q_j=\{n_j,\ldots,n_j+|F_j|-1\}$ and $E=\{e_1,\ldots,e_i\}\}$. With the same construction method, one can easily prove that there exists $H_i\in\mathcal{F}$ such that:

(1) for any k-tuple $j_1,\ldots,j_k\in\{1,\ldots,i\}$ with at least 2 distinct elements we have
$$\delta_F(a_{j_1},\ldots,a_{j_k})=a_\alpha \implies \delta_{H_i}(e_{j_1},\ldots,e_{j_k})=e_\alpha,$$
(2) for any k-tuple $x_1,\ldots,x_k$ with $\{x_1,\ldots,x_k\}\not\subseteq Q_j$ (for all j) and $\{x_1,\ldots,x_k\}\not\subseteq E$ we have
$$\delta_{H_i}(x_1,\ldots,x_k)=\max\{x_j : x_j\notin E\},$$
(3) $F^{m_0}\sqsubseteq H_i^{m_0}$.

Now suppose $H_i\in\mathcal{U}$ and consider some $G_u\in\mathcal{U}$ with only 2 states (but with a neighbourhood possibly larger than k). Since by hypothesis $G_u\preceq H_i$, proposition 5.1 implies that there are parameters m,t such that $G_u\sqsubseteq H_i{}^{<m,t>}$. Denote by Σ the set of configurations of H_i where the simulation occurs. First, we have $\Sigma\not\subseteq Q_j^{\mathbb{Z}}$ (for all j) since otherwise it would imply that $F_j\in\mathcal{U}$. Moreover, if we suppose $\Sigma\not\subseteq E^{\mathbb{Z}}$, then there is some configuration $c\in\Sigma$ and some position $z\in\mathbb{Z}$ such that $\{c_z,\ldots,c_{z+k-1}\}\not\subseteq Q_j$ and $\{c_z,\ldots,c_{z+k-1}\}\not\subseteq E$. Then condition 2 of the definition of H_i applies at position z. Intuitively, starting from position z, a region of states all in Q_j for the same j will grow unless it meets some state in $Q_{j'}$ with $j'>j$ in which case a larger $Q_{j'}$ region appears. Applying this reasoning until the maximal j is reached, it is straightforward to show that there is t_0 such that $\forall t_+\geq t_0$ the configuration $d=H_i^{t_+}(c)$ has for some $z'\in\mathbb{Z}$ and some j the property:
$$\forall z'', z'\leq z''\leq z'+2m \;:\; d(z'')\in Q_j.$$
It means that some state of G_u is simulated by group of states of H_i all in Q_j for the same j. This implies either $\Sigma\subseteq Q_j$ or that G_u has a spreading state (a state x such that each cell with x in its neighbourhood turns into state x). The two cases being contradictory with hypothesis ($F_j\notin\mathcal{U}$ and $G_u\in\mathcal{U}$ and has only two states), we conclude that $\Sigma\subseteq E^{\mathbb{Z}}$.

Therefore, by condition 3 of the definition of H_i, we have $G_u \sqsubseteq F^{<mm_0,t>}$ and thus $F \in \mathcal{U}$ which concludes the proof. ∎

Using results, one can prove the following theorem showing that the intersection with $\mathcal{U}$ of families $\mathcal{K}$, $\mathcal{MAJ}$, $\mathcal{MIN}$ and any unambiguous $\mathcal{F}$ associated to Γ_a are non-trivial.

Theorem 5.4. *There exists r_0 such that for any fixed Von Neumann neighbourhood of radius $r \geq r_0$, and for any family $\mathcal{F}$ among $\mathcal{K}$, $\mathcal{MIN}$, $\mathcal{MAJ}$ or an unambiguous family associated to Γ_a, then we have $d_{\mathcal{F}}(\mathcal{U}) = 1$.*

Moreover if $\mathcal{F}$ is neither $\mathcal{MIN}$ nor $\mathcal{MAJ}$, we have also:

- *$\mathcal{U}$ is undecidable in $\mathcal{F}$,*
- *for any $F \in \mathcal{F} \setminus \mathcal{U}$ there is $G \in \mathcal{F} \setminus \mathcal{U}$ with $F \preceq G$ but $G \not\preceq F$.*

Proof. The first part of the theorem follows from propositions 5.2 and 5.3, from the existence of universal captive CA (proven in [17] for $r \geq 7$) and from corollary 4.5.

The second part is proved in theorem 2 and corollary 3 of [17] for the family $\mathcal{K}$. It generalises to unambiguous families associated to Γ_a by propositions 5.3 above. ∎

Much less is known about the structure of $\preceq$ concerning reversible CA. However, as we will show below, there exists reversible-universal captive CA. This naturally raises the question (unanswered in this paper) of the density of reversible-universal CA among captive reversible CA. The existence of reversible-universal captive CA relies on the existence of reversible-universal CA of a special kind, equipped with an unalterable state.

Proposition 5.5. *There exists a reversible-universal F such that F and its inverse F^{-1} possesses a common wall state, i.e. a state q such that any cell in state q remains in state q under both the action of F and F^{-1}.*

Proof. Consider any reversible-universal CA G (see [7] for an existence proof) of state set Q_G and radius r. Let q be an additional state not in Q_G and let $Q = Q_G \times Q_G \cup \{q\}$ (it is straitghtforward to translate this state set into a subset of $\mathbb{N}$, but we don't for clarity). We will see any configuration $c \in Q^{\mathbb{Z}}$ as the disjoint union of configurations from $Q_G^{\mathbb{Z}}$ corresponding to zones between occurrences of state q. A finite zone between two occurrences of q will be seen as a torus of states from Q_G, semi-infinite zones as bi-infinite configuration in $Q_G^{\mathbb{Z}}$ and bi-infinite zones as pair of configurations in $Q_G^{\mathbb{Z}}$. We formalise this through functions $L^{\uparrow}$, $L^{\downarrow}$, $R^{\uparrow}$, $R^{\downarrow}$ which give the state of the "logical" neighbours of a "logical" cell (according to the previous description): L/R stand for left/right neighbours and $\uparrow/\downarrow$ for up/down layers (in a zone without q, cells contain 2 layers of "logical" cells).

Formally, for any $z \in \mathbb{Z}$ and $c \in Q^{\mathbb{Z}}$ such that $c(z) \neq q$ these functions are defined by:

$$L^{\uparrow}(z,p) = \begin{cases} \epsilon & \text{if } p = 0 \\ L^{\uparrow}(z-1,p-1)\cdot a & \text{if } c(z-1) = (a,b) \in Q_G^2 \\ L^{\downarrow}(z,p-1)\cdot \beta & \text{if } c(z-1) = q \text{ and } c(z) = (\alpha,\beta) \end{cases}$$

$$L^{\downarrow}(z,p) = \begin{cases} \epsilon & \text{if } p = 0 \\ L^{\downarrow}(z+1,p-1)\cdot b & \text{if } c(z+1) = (a,b) \in Q_G^2 \\ L^{\uparrow}(z,p-1)\cdot \alpha & \text{if } c(z+1) = q \text{ and } c(z) = (\alpha,\beta) \end{cases}$$

$$R^{\uparrow}(z,p) = \begin{cases} \epsilon & \text{if } p = 0 \\ a\cdot R^{\uparrow}(z+1,p-1) & \text{if } c(z+1) = (a,b) \in Q_G^2 \\ \beta\cdot R^{\downarrow}(z,p-1) & \text{if } c(z+1) = q \text{ and } c(z) = (\alpha,\beta) \end{cases}$$

$$R^{\downarrow}(z,p) = \begin{cases} \epsilon & \text{if } p = 0 \\ b\cdot R^{\downarrow}(z-1,p-1) & \text{if } c(z-1) = (a,b) \in Q_G^2 \\ \alpha\cdot R^{\uparrow}(z,p-1) & \text{if } c(z-1) = q \text{ and } c(z) = (\alpha,\beta) \end{cases}$$

Now define F on any $c \in Q^{\mathbb{Z}}$ by:

$$\big(F(c)\big)(z) = \begin{cases} q & \text{if } c(z) = q, \\ \big(\delta_G(L^{\uparrow}(z,r)c(z)R^{\uparrow}(z,r)), \delta_G(L^{\downarrow}(z,r)c(z)R^{\downarrow}(z,r))\big) & \text{else.} \end{cases}$$

F is a well-defined CA since $L^{\uparrow}(z,r)$, $R^{\uparrow}(z,r)$, $(L^{\downarrow}(z,r)$ and $R^{\downarrow}(z,r)$ depend only on the $2r+1$ cells surrounding z. It is clear from the definition that $G \preceq F$ since F behave like the product of G and its symmetric on configuration without q.

To show that F is reversible, it is sufficient to notice that applying the same construction to G^{-1}, one gets a CA F' such that for any $c \in Q^{\mathbb{Z}}$: $F'\big(F(c)\big) = c$. ∎

Proposition 5.6. *There exists a reversible-universal captive CA.*

Proof. Let F be any CA of radius r, state set $Q_F = \{q_1,\dots,q_n\}$ and having a wall state q. One defines the captive CA F_κ on state set $Q = Q_F \cup \{L,R\}$ with radius $r' = (n+3)(r+1)$ as follows. Let $u = Lq_1q_2\cdots q_nR$. Any word $w = w_{-r'}\cdots w_0\cdots w_{r'}$ of length $2r'+1$ over Q such that $w_0 \in Q_F$ can be written in the form:

$$w = \underbrace{v_lux_iux_{i-1}u\cdots u}_{r'}\,x_0\,\underbrace{ux_1u\cdots x_juv_r}_{r'}$$

where $x_p \in Q_F$ for all p and i and j are maximal for this form. To the word w we associate the word $\pi(w) = q^{r-i}x_i\cdots x_0\cdots x_jq^{r-j}$. Now we define F_κ by:

$$\delta_{F_\kappa}\big(e_{-r'},\dots,e_{r'}\big) = \begin{cases} e_0 & \text{if } e_{-(n+2)}\cdots e_{n+2} \notin uQ_Fu, \\ \delta_F\big(\pi(e_{-r'}\cdots e_{r'})\big) & \text{else.} \end{cases}$$

For any $c \in Q^{\mathbb{Z}}$, let $A(c) = \{z \in \mathbb{Z} : c(z)\cdots c(z+n+1) = u\}$. It is straightforward from the definition of F_κ to check that $A(c) = A\big(F_\kappa(c)\big)$. Therefore we also have $B(c) = B\big(F_\kappa(c)\big)$ where $B(c) = \{z \in \mathbb{Z} : c(z-(n+2))\cdots c(n+2) \in uQ_Fu\}$. Now, if we apply this construction to F and F^{-1} obtained by proposition 5.5, we have from the above remark that $F_\kappa \circ F_\kappa^{-1}$ is the identity and therefore F_κ is reversible.

Finally, one can easily check that $F \preceq F_\kappa$ by considering configurations of the form ${}^{\omega}(uQ_F)^{\omega}$ and the proposition follows. ∎

6. Perspectives

This paper leaves many question unanswered. First, although it was not directly addressed in this paper, determining the density of $\mathcal{U}$ in the whole set of CA remains a completely open problem of prior interest.

Concerning the amalgamation operations themselves, it would be interesting to follow [4] and [5] and define new amalgamation operations where the way of completing transition tables is controlled by a two-state CA. For such amalgamation operations (and for Γ_g also) we believe that the zero-one law is still valid (at least when the set of generators is unambiguous). However, for the amalgamation of J. Mazoyer *et al.*, knowing when and how universality can be achieved starting from non-universal generators seems very difficult.

More generally, for any amalgamation operation, the associated family has only a finite set of nilpotent. Therefore nilpotency is a decidable property in the family and many classical undecidable problems are to be reconsidered when restricted to such families. Among them, we think that reversibility and surjectivity in dimension 2 and properties of limit sets are particularly intersting.

Concerning reversible CA in dimension 1, we wonder whether it is possible to define a non-trivial amalgamation family made of reversible CA only and for which density of reversible-universality is 1. To that extent, we remark that no majority CA is reversible and there are reversible-universal captive CA, so amalgamation families lying between $\mathcal{MAJ}$ and $\mathcal{K}$ are worth being considered.

Finally, even if some amalgamation families share many properties concerning the class $\mathcal{U}$, there is no reason why typical CA from two different families should have a similar dynamical behaviour. A possible formalization of this could be to study the μ-limit sets of typical CA from different families. For instance, one can establish that μ-limit sets of majority CA consists in fixed-point configuration only (proposition omitted from the present paper due to lack of space). We believe that non-trivial properties of μ-limit sets of typical CA from $\mathcal{K}$ or other families could be established.

References

[1] J. Albert and K. Čulik II. A simple universal cellular automaton and its one-way and totalistic version. *Complex Systems*, 1:1–16, 1987.

[2] E. R. Banks. Universality in cellular automata. In *Eleventh Annual Symposium on Switching and Automata Theory*, Santa Monica, California, 1970. IEEE.

[3] M. Cook. Universality in elementary cellular automata. *Complex Systems*, 15:1–40, 2004.

[4] M. Delorme and J. Mazoyer. Cellular automata and amalgamation. AUTOMATA'02, 8th IFIP WG1.5 workshop, 2002.

[5] M. Delorme and J. Mazoyer. Private communications. 2005.

[6] B. Durand and Z. Róka. *Cellular Automata: a Parallel Model*, volume 460 of *Mathematics and its Applications.*, chapter The game of life:universality revisited., pages 51–74. Kluwer Academic Publishers, 1999.

[7] J. Durand-Lose. Intrinsic universality of a 1-dimensional reversible cellular automaton. In *Annual Symposium on Theoretical Aspects of Computer Science*, 1997.

[8] J. Kari. The Nilpotency Problem of One-dimensional Cellular Automata. *SIAM Journal on Computing*, 21:571–586, 1992.

[9] J. Kari. Reversibility and Surjectivity Problems of Cellular Automata. *Journal of Computer and System Sciences*, 48(1):149–182, 1994.

[10] Bruno Martin. A universal cellular automaton in quasi-linear time and its S-m-n form. *Theoretical Computer Science*, 123(2):199–237, 1994.

[11] Cristopher Moore. Majority-vote cellular automata, ising dynamics, and p-completeness. *J. stat. phys.*, 88(3-4):795–805, 1997.

[12] N. Ollinger. *Automates Cellulaires : structures*. PhD thesis, École Normale Supérieure de Lyon, dcembre 2002.

[13] N. Ollinger. The quest for small universal cellular automata. In *International Colloquium on Automata, Languages and Programming*, pages 318–330. Lecture Notes in Computer Science, 2002.

[14] N. Ollinger. The intrinsic universality problem of one-dimensional cellular automata. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 632–641. Lecture Notes in Computer Science, 2003.

[15] G. Richard. A particular universal cellular automaton. `oai:hal.archives-ouvertes.fr:hal-00095821_v1`, 2006.

[16] G. Theyssier. Captive cellular automata. In *Mathematical Foundations of Computer Science*, pages 427–438. Lecture Notes in Computer Science, 2004.

[17] G. Theyssier. How common can be universality for cellular automata? In *Annual Symposium on Theoretical Aspects of Computer Science*, 2005.

Journées Automates Cellulaires 2008 (Uzès), pp. 195-203

GOTO'S CONSTRUCTION AND PASCAL'S TRIANGLE: NEW INSIGHTS INTO CELLULAR AUTOMATA SYNCHRONIZATION

JEAN-BAPTISTE YUNÈS [1]

[1] LIAFA, Université Paris Diderot/CNRS
Case 7014
75205 Paris Cedex 13. France.
E-mail address: Jean-Baptiste.Yunes@liafa.jussieu.fr
URL: http://www.liafa.jussieu.fr/~yunes/

Abstract. Here we present a new non-recursive minimal-time solution to the Firing Squad Synchronization Problem which does not use any recursive process. In 1962, E. Goto designed an iterative algorithm which uses Minsky-McCarthy's solutions to synchronize in minimal-time. Our solution does not use any standard recursion process, only some "fractal computation", making it a purely iterative synchronization algorithm.

Introduction

The firing squad synchronization problem (FSSP for short) has been the subject of many studies since 1957 when Myhill stated it and Moore reported it (see [Mo64]). We can state the problem as follows:

> Does there exist a finite automaton such that a chain of n (whatever n could be) such automata would be synchronized at some time $\mathcal{T}(n)$ after being initiated at time $t = 0$? Each automaton is connected with its two neighbors and is assumed to be structurally independent of the number n. The synchronization is a configuration such that each automaton is in a so-called firing state which was never used before time $\mathcal{T}(n)$ and the ignition configuration is a configuration such that every automaton but the first one of the chain is in a quiescent state.

Besides the fact that numerous papers were published about it and many different solutions were designed to solve the problem in various conditions, one of the very first solution made by Goto remained mythical for a long time. His courses notes are not available and Goto has not published his solution elsewhere. Many years later, Umeo (see [Um96]) was the first who tried to reconstruct it as he was able to talk to Goto himself who then gave him some old incomplete drawing. After that, Mazoyer (see [Ma98]) made a possible reconstruction of it but did not published it.

Key words and phrases: parallel computations, synchronization, firing squad, cellular automata.

In this paper we do not try to strictly reconstruct Goto's solution but to use his main idea to build a new minimal-time solution with the following interesting characteristics in mind:

- the iterative process ensures that we do not need a complex discretization process;
- the set of signals used is small; we will see that an implementation will give us the opportunity to use only two different signals (slope 1 and slope 3);
- there is only one "cut" of the line;
- we want to obtain a solution whose energy consumption is lower than n^2;

1. The schema

Figures 4, 5 , 6 illustrate the overall mechanism involved. Roughly speaking, we can say that the main idea, due to Goto (see [Go62]) is to split the line into successive sublines the lengths of which are a sequence of powers of 2 (2^0, 2^1, 2^2, etc.), from the left and from the right. We will use this iterative decomposition to place some minimal-time firing squad. Of course it is possible to use any minimal-time solution, even the one we are constructing as Gerken did, but we choose to use some specific solutions able to efficiently synchronize powers of 2 - see [Yu08].

As it is not always possible to cover the line with such left and right sublines something must be built in the middle to ensure that the residue is also correctly synchronized, either ensuring some overlapping or filling the hole.

Different constructions are involved:

- splitting the line into successive sublines of length powers of 2 from the left;
- starting appropriate minimal-time solutions on the left sublines;
- splitting the line into successive sublines of length powers of 2 from the right;
- starting appropriate minimal-time solutions on the right sublines;
- filling the *empty space* in the middle of the line when necessary.

In the following we will always consider that cells of the chain are numbered from 0 (the left cell) to $n-1$ (the right cell) and that the time starts at 0.

2. Splitting the line into successive sublines of length 2^i from the left

Splitting a line into successive power of 2 is easy and is illustrated in Figure 1(a). Suppose that length l has already been constructed in space, *i.e.*, the distance between the abscissas of the two sites P'_i and P'_{i-1} is l.

From P'_i two signals are issued. The first, of slope 1, goes to the left until it meets the previous stationary signal issued from P'_{i-1}, then bounces back in the reverse direction until it meets the signal of slope 2 issued from P'_i to the right. That crossing point P'_{i+1} is the new starting point of the next construction. If we start with $l = 1$, then it is easy to see that all powers of 2 are successively constructed in space.

The previous construction is due to Goto, but for different reasons that we will explain later, we need to also use another construction illustrated in Figure 1(b). Suppose that length l has been constructed and that P_i and P'_i are located on the same vertical and distant by l. Then, the meeting of the signal of slope 1 issued from P'_i and the signal of slope $\frac{3}{2}$ issued from P_i is P_{i+1} and is exactly at distance $2l$ (in space) from P_i. At the same time if we start a signal of slope 2 from P'_i, it meets the stationary signal issued from P_{i+1}

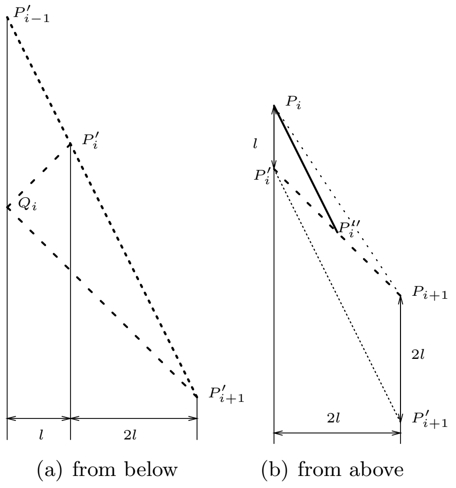


Figure 1: Constructing P_i's, P'_i's

at point P'_{i+1} distant from P_{i+1} by $2l$. Then again it is easy to see that if we start with $l = 1$, all powers of 2 are successively constructed in space.

We shall also need the midle P''_i of P'_iP_{i+1} which in obtained by starting a signal of slope 2 from P_i.

Starting from $P'_1 = [1,1]$ then, with the preceding constructions, one can see that, for all $i \geq 2$, the constructed points have the following coordinates:

$$\begin{aligned} P_i &= [\,2^i - 1\ ,\ 3.(2^{i-1} - 1)\,] &(2.1)\\ P'_i &= [\,2^i - 1\ ,\ 2^{i+1} - 3\,] &(2.2)\\ P''_i &= [\,3.2^{i-1} - 1\ ,\ 5.2^{i-1} - 3\,] &(2.3)\\ Q_i &= [\,2^{i-1} - 1\ ,\ 5.2^{i-1} - 3\,] &(2.4)\end{aligned}$$

The previous signals and the above equations are more easily viewed with the help of Pascal's triangle modulo 2 (which is obtained via Wolfram's rule 60) as one can see in Figure 2.

3. Synchronizing left sublines

To synchronize the cells of the left constructed sublines, one can use the schema illustrated by Figure 3(a).

Every stationary signal issued from P_i meets the return of the main signal (a line of equation $y = 2n - x$) at $S_i = [2^i - 1, 2n - 2^i + 1]$ where a minimal-time solution can be started with an initiator at right using the stationary signal issued from S_{i-1} as a border.

4. Splitting the line into successive sublines of length 2^i from the right and synchronizing them

The construction is illustrated by Figure 3(b). As one can see, we use the left construction to build the right sublines. From every Q_i a signal of slope 1 is started to the right

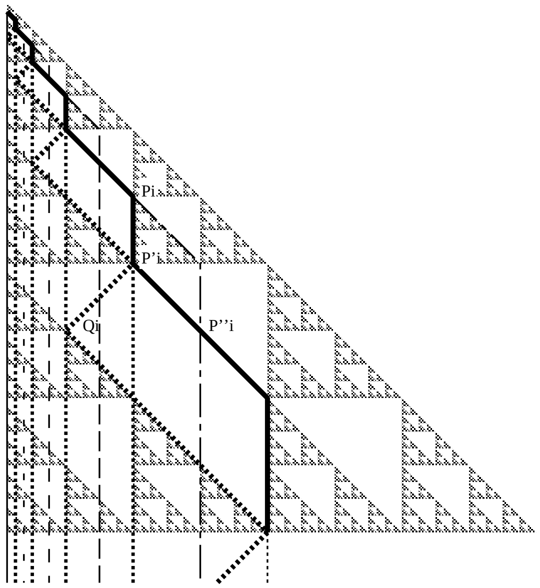


Figure 2: Rule 60 helps

until it meets the return of the main signal, then at that point a stationary signal is set up which will meet the symmetric counterpart of the return of the main signal issued from the middle of the line at a point named S'_i. From each of those points, a minimal-time solution can be started with an initiator at left, using the stationary signal issued from S'_{i-1} as a border.

5. Filling the empty space

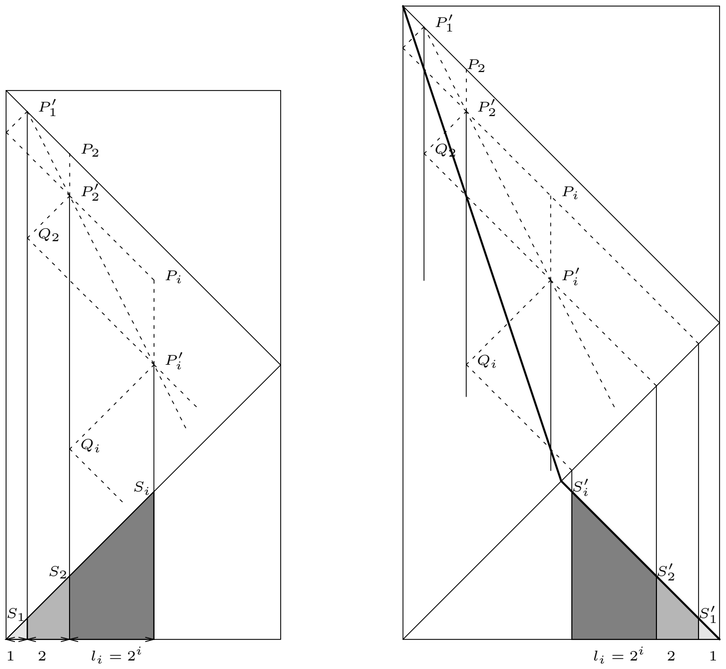
Depending on the length of the line it is necessary to carefully consider what happens in the middle of the line. Points S_i and S'_i are symmetric relative to the middle of the line. Depending whether S_i, is left or right to S'_i, the last left and right sublines overlap or not. A simple analysis shows that there exists three different cases to consider about the position of points P, P', P'' relatively to the return of the main signal (signal of slope 1 issued from the left cell which bounces back from the right border).

Whatever be n, there exists i such that one and only one of the following cases occurs:

(1) P_i, P'_i and P''_i are all constructed before the main signal has returned and P_{i+1} is not;
(2) P_i and P'_i are constructed before the main signal has returned and P''_i is not;
(3) P_i is constructed before the main signal has returned and P'_i is not.

5.1. Case 1: P, P' and P" before

Figure 4 illustrates the case where there is an index i such that P''_i has been built before the main signal has returned and it is not the case for P_{i+1}. With the help of Eqs 2.1



(a) from left (b) from right

Figure 3: Synchronizing sublines

and 2.3, these two conditions lead to:

$$\begin{cases} 5.2^{i-1} - 3 & \leq & 2n - 3.2^{i-1} + 1 \\ 2n - 2^{i+1} + 1 & < & 3.(2^i - 1) \end{cases}$$

then to

$$\begin{cases} 2^{i+1} - 2 & \leq & n \\ n & < & 2^{i+1} + 2^{i-1} - 2 \end{cases}$$

and the following equation holds

$$2^{i+1} - 2 \leq n < 2^{i+1} + 2^{i-1} - 2 \tag{5.1}$$

In that case synchronization is achieved by constructing the following firing squads:

- i minimal-time FS are started on sites $S_k = [2^k - 1{,}2n - 2^k - 1]$ $(1 \leq k \leq i)$;
- i minimal-time FS are started on sites $S'_k = [n - 2^k - 1{,}2n - 2^k - 1]$ $(1 \leq k \leq i)$. A special case has to be considered for S'_i as P'_{i+1} has not been built by definition. To build S'_i it is sufficient to compute $Q_i = [2^{i-1} - 1{,}5.2^{i-1} - 3]$ and its meeting point with the return of main signal, *i.e.* the meeting point of $L \,:\, y = x + 2^{i+1} - 2$ and $L' \,:\, y = 2n - x$ which has abscissa $n - 2^i + 1$;
- an additional minimal-time FS can be started on site S_i, propagates to the right and uses the stationary signal issued from P''_i as its end-of-line.

For this schema to work we must verify that:

- S'_i can always be built on time (it must appear before the middle of the line)

- S'_i must appear at the left of P''_i, so that all cells in between S_i and S'_i can be synchronized by an appropriate FS.

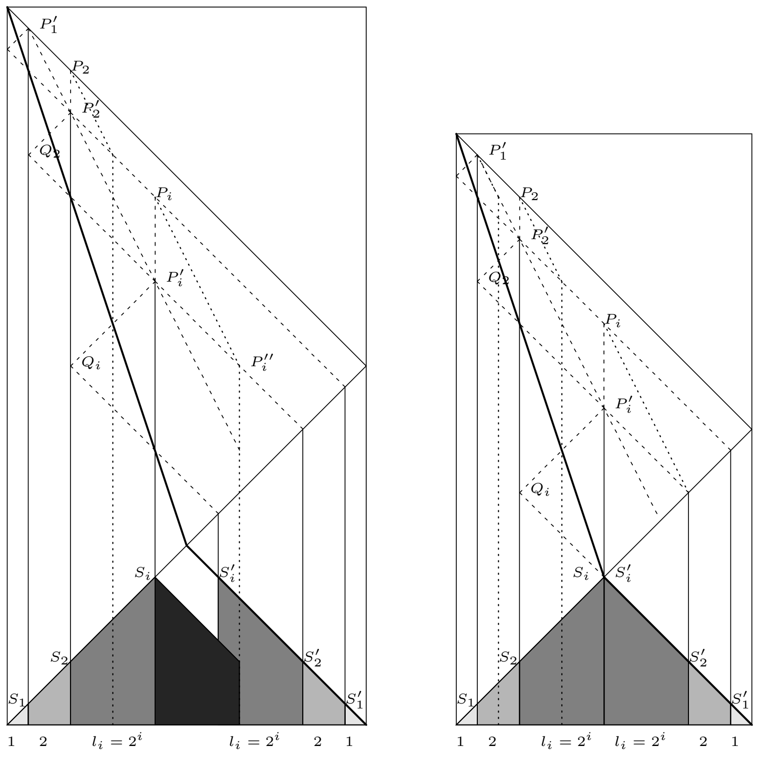
We know that S'_i has coordinates $n - 2^i + 1$ and that P''_i has abscissa $5.2^{i-1} - 3$ then we must have:

$$\begin{cases} \frac{n}{2} & \leq & n - 2^i + 1 \\ n - 2^i + 1 & < & 5.2^{i-1} - 3 \end{cases}$$

which is

$$\begin{cases} 2^{i+1} - 2 & \leq & n \\ n & < & 7.2^{i-1} - 4 \end{cases}$$

which is implied by Equation 5.1



(a) general case (b) limit case

Figure 4: P, P' and P" before

5.2. Case 2: P and P' stricly before, P" after

Figure 5 shows what is constructed when there is an index i such that P'_i has been built strictly before the return of the main signal and that it is not the case for P''_i. With the help of Equations 2.2 and 2.3, these conditions lead to:

$$\begin{cases} 2^{i+1} - 3 & < & 2n - 2^i + 1 \\ 2n - 3.2^{i-1} + 1 & < & 5.2^{i-1} - 3 \end{cases}$$

which gives

$$\begin{cases} 3.2^{i-1}-2 & < & n \\ n & < & 2^{i+1}-2 \end{cases}$$

and the following equation holds

$$3.2^{i-1}-2 < n < 2^{i+1}-2 \tag{5.2}$$

In that case synchronization is achieved by the following constructions:

- i minimal-time FS are started on sites $S_k = [2^k-1{,}2n-2^k-1]$ $(1 \le k \le i)$;
- $i-1$ minimal-time FS are started on sites $S'_k = [n-2^k-1{,}2n-2^k-1]$ $(1 \le k < i)$;
- a minimal-time FS is started from S_i propagating to the right and using the stationary signal issued from P''_i as its end-of-line.

For this schema to work some conditions must be verified:

- P''_i must be constructible;
- S'_{i-1} must be at the left of P''_i so that the appropriate FS started at site S_i synchronizes at least the cells in between S_i and S'_{i-1}.

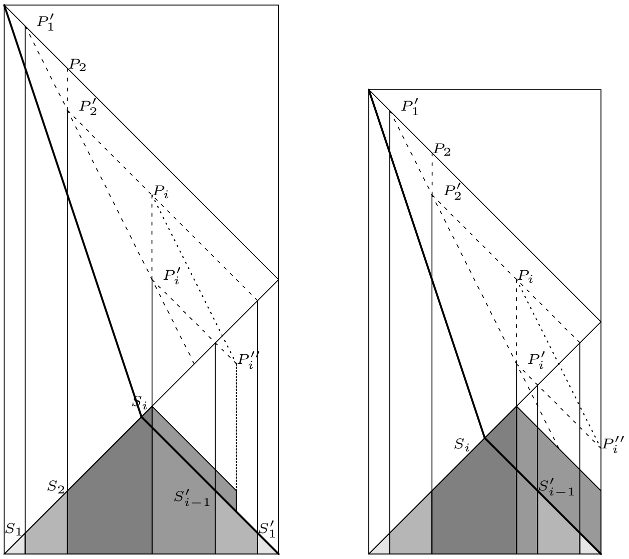
We know that the abscissa x of S'_{i-1} is solution of $2n-x = x-2^i+1+2^{i+1}-3$, so that $x = n-2^{i-1}+2$. So we must have:

$$\begin{cases} 3.2^{i-1}-1 & \le & n \\ n-2^{i-1}+2 & \le & 3.2^{i-1}-1 \end{cases}$$

which gives

$$\begin{cases} 3.2^{i-1}-1 & \le & n \\ n & \le & 2^{i+1}-3 \end{cases}$$

which is exactly the condition of Equation 5.2.



(a) general case (b) limit case

Figure 5: P, P' before, P" after

5.3. Case 3: P before, P' and P" after

Figure 6 illustrates the construction when there is an index i such that P_i has been built before the return of the main signal and that it is not the case for P'_i. Then with the help of Equations 2.1 and 2.2, we have:

$$\left\{\begin{array}{rcl} 3.(2^{i-1}-1) & \leq & 2n-2^i+1 \\ 2n-2^i+1 & \leq & 2^{i+1}-3 \end{array}\right.$$

thus

$$\left\{\begin{array}{rcl} 2^i+2^{i-2}-2 & \leq & n \\ n & \leq & 3.2^{i-1}-2 \end{array}\right.$$

and the following equation holds

$$2^i+2^{i-2}-2 \leq n \leq 3.2^{i-1}-2 \tag{5.3}$$

In that case synchronization is achieved by constructing the following firing squads:

- i minimal-time FS are started on $S_k = [2^k-1{,}2n-2^k-1]$ $(1 \leq k \leq i)$;
- $i-1$ minimal-time FS can be started on $S'_k = [n-2^k-1{,}2n-2^k-1]$ $(1 \leq k < i)$. Note that S'_{i-1} is built by a special process issued from Q_{i-1} as done in case 1.

For all this to work correctly, some conditions must be verified:

- S'_{i-1} must appear on the left of S_i, such that an every cell will be synchronized.
- S'_{i-1} must appear on time (before the middle cut of the line).

We know that the abscissa x of S'_{i-1} is solution of $2n-x = x-2^{i-2}+1+5.2^{i-2}-3$ then that $x = n-2^{i-1}+1$. So we must have:

$$\left\{\begin{array}{rcl} n-2^{i-1}+1 & \leq & 2^i-1 \\ \frac{n}{2} & \leq & n-2^{i-1}+1 \end{array}\right.$$

which gives

$$\left\{\begin{array}{rcl} n & \leq & 3.2^{i-1}-2 \\ 2^i-2 & \leq & n \end{array}\right.$$

which is implied by Equation 5.3.

As for every integer n , there exists i such that $2^i-2 \leq n < 2^{i+1}-2$, from Equations 5.1, 5.2, 5.3, this proves the main result of this paper.

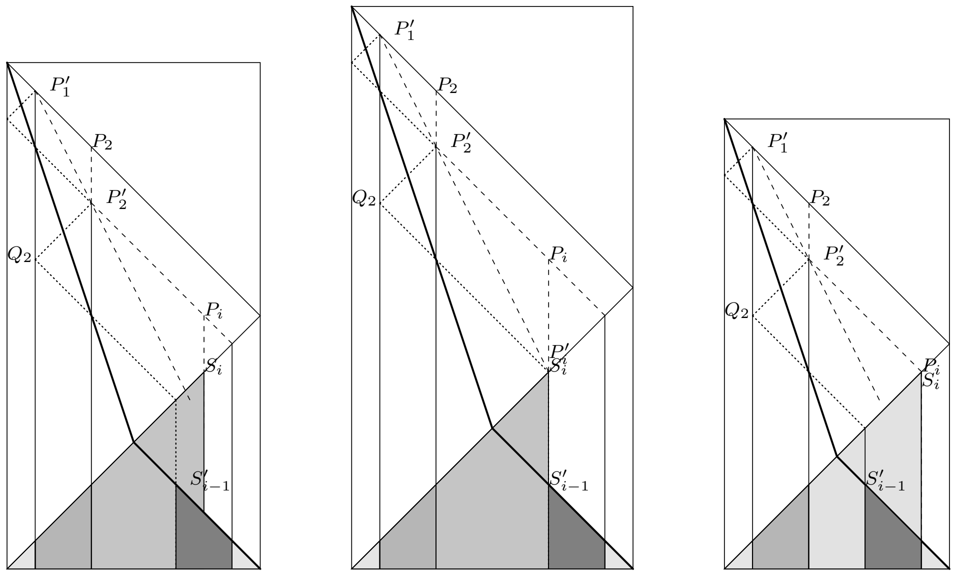
Theorem 5.1 (Yunès). *The schema synchronizes every line of length $n \in \mathbb{N}$.*

6. Conclusion

A strict implementation of the preceding schema is possible but we would like to show how many interesting optimizations can be done.

First we can remark that if any minimal-time solution can be used to synchronize the sublines, every subline has a length which is a power of 2. Then according to Yunès and Umeo (see [Yu08] and [Um07]), we know that it is possible to synchronize a line of length 2^k with only 4 states, such solutions are algebraic and do not use any signal. Thus using such a construction will certainly lower down the total number of states.

But more than this, if we use one of these 4-state solutions then we can use them as the support for the construction of all the interesting points P, P', P'' and Q as one can see in Figure 2.



(a) general case (b) limit case (c) limit case

Figure 6: P before, P' and P" after

Now one can remark that there are only two kind of signals: slope 1 and slope 3. And the signal of slope 3 is only used to cut the main line into two equals parts. We actually do not know if an explicit construction of this signal is necessary.

Besides the fact that such a schema is the very first one, one can observe that it has many interesting characteristics which probably nobody never thought about.

References

[Ge87] Hans-D. Gerken. *Über Synchronizations - Probleme bei Zellularautomaten.* Diplomarbeit, Institut für Theoretische Informatik, Technische Universität Braunschweig, **1987**.

[Go62] Eiichi Goto. *A Minimum Time Solution of the Firing Squad Problem.* Courses Notes for Applied Mathematics 298, Havard University, pp. 52–59, **1962**.

[Ma98] Jacques Mazoyer. *A minimal-time solution to the FSSP without recursive call to itself and with bounded slope of signals.* Unpublished draft, private communication, **1998**.

[Mo64] Edward E. Moore. *Sequential Machines, Selected papers.* Addison-Wesley, **1964**.

[Um96] Hiroshi Umeo. *A Note on Firing Squad Synchronization Algorithms.* IFIP Cellular Automata Workshop 96, Schloss Rauischholzhausen, Giessen, pp. 65, **1996**.

[Um02] Hiroshi Umeo, Masaya Hisakoa, Takashi Sogabe. *An Investigation into Transition Rule Sets for Optimum-time Firing Squad Synchronization Algorithms on One-Dimensional Cellular Automata.* Interdisciplinary Information Sciences, Vol. 8, No. 2, pp. 207–217, **2002**.

[Um07] Hiroshi Umeo, Naoki Kamikawa. *A 4-state solution to the firing squad based on Wolfram's rule 150.* Private communication, **2007**.

[Yu08] Jean-Baptiste Yunès. *A 4-stats Algebraic Solution to Linear Cellular Automata Synchronization.* Information Processing Letters. To appear, **2008**. doi: 10.1016/j.ipl.2008.01.009.

Journées Automates Cellulaires 2008 (Uzès), pp. 204-215

QUANTIZATION OF CELLULAR AUTOMATA

PABLO ARRIGHI [1] AND VINCENT NESME [2]

[1] Université de Grenoble
LIG, 46 Avenue Félix Viallet
38031 Grenoble Cedex FRANCE
E-mail address: pablo.arrighi@imag.fr

[2] Technische Universität Braunschweig
IMaPh, Mendelssohnstraße 3
38106 Braunschweig DEUTSCHLAND
E-mail address: vincent.nesme@tu-bs.de

Abstract. Take a cellular automaton, consider that each configuration is a basis vector in some vector space, and linearize the global evolution function. If lucky, the result could actually make sense physically, as a valid quantum evolution; but does it make sense as a quantum cellular automaton? That is the main question we address in this paper. In every model with discrete time and space, two things are required in order to qualify as a cellular automaton: invariance by translation and locality. We prove that this locality condition is so restrictive in the quantum case that every quantum cellular automaton constructed in this way — i.e., by linearization of a classical one — must be reversible. We also discuss some subtleties about the extent of nonlocality that can be encountered in the one-dimensional case; we show that, even when the quantized version is non local, still, under some conditions, we may be unable to use this nonlocality to transmit information nonlocally.

Introduction

After some tries [9, 4, 5, 1] at defining and studying quantum cellular automata, it is now believed to be fairly well understood how reversible quantum cellular automata (RQCA) should be defined, and what their basic properties are. As with classical cellular automata (CA), there are two levels on which RQCA are defined: as local transition functions or as global evolutions. The definition of RQCA proposed in [8] focuses only on the properties of the global evolution, based on the two essential points of invariance by translation and locality. It was also proved in the same paper that each reversible cellular automaton could be "quantized" in a natural way, and the result would be a RQCA. Furthermore, it was proved that RQCA can be implemented with local means, thereby reinforcing the parallel with CA; this was first done in the one-dimensional case [8, 2], the result being later

2000 ACM Subject Classification: F.1.1.
Key words and phrases: cellular automata, quantum cellular automata, open cellular automata, quantization, locality, localization.

extended to the general case [3]. Also, they involve no measurement procedure; the global evolution of a RQCA can be described by a unitary operator, while its decomposition as layers of local operations consists only of small unitary transformations.

We would like now to extend this framework to include cases where the global evolution is no longer described by a unitary operator, but by an isometry. This would be the first step in the investigation of nonreversible quantum cellular automata (NRQCA). The main problem with this topic, nowadays, is that there is no practical definition for such things. Our aim is to provide such a definition and work out the basic properties of NRQCA. Invariance by translation and locality as defined in [2] are still properties that NRQCA should obviously have. In this paper we will ask and answer this question: when does the quantization of a CA have these properties? Since the translational invariance comes freely, the real question is: when is the quantization local?

Section 1 will be devoted to the mandatory definitions. We will be quick as we assume the reader is familiar with the basics of CA and somewhat familiar with quantum computing. We then show with theorem 2.1 that the locality — more precisely, the *uniform* locality, cf. definition 1.7 — of the quantization is equivalent to reversibility, therefore extending the results presented in [8, 2], and proving that no NRQCA can be constructed in this way. In Section 3, we discuss the one-dimensional case. We show that, in some cases, even if the quantization is not uniformly local, it can still be local in a weaker sense which forbids some kinds of long-distance communications.

1. Definitions

We will now introduce the basic definitions of quantum cellular automata. For technical reasons, we will work mainly with finite configurations. This is because they are countable, as opposed to infinite configurations, and we want to have vector spaces of countable dimension, so as to simplify the formalism of [8]. This distinction between finite and infinite configurations is not so important, as was shown in [2]; anyway, we are only interested in locality conditions for quantizations of CA. We do not restrict the dimension of the space, which will be some positive integer d. We denote q the quiescent state, and Σ the rest of the alphabet, assuming $q \notin \Sigma$; the union of Σ and $\{q\}$ is denoted $q\Sigma$. The sets of finite configurations is denoted $\mathcal{C}_f$; it contains the elements of $(q\Sigma)^{\mathbb{Z}^d}$ that are equal to q almost everywhere on $\mathbb{Z}^d$.

Whilst configurations hold the basic states of an entire line of cells, and hence denote the possible basic states of the entire QCA, the global state of a QCA may well turn out to be a superposition of these. The following definition works because $\mathcal{C}_f$ is a countably infinite set.

Definition 1.1 (Superpositions of configurations).
Let $\mathcal{H}_{\mathcal{C}_f}$ be the Hilbert space of configurations, defined as follows. To each finite configuration c is associated a unit vector $|c\rangle$, such that the family $(|c\rangle)_{c\in\mathcal{C}_f}$ is an orthonormal basis of $\mathcal{H}_{\mathcal{C}_f}$. A *superposition of configurations* is then a unit vector in $\mathcal{H}_{\mathcal{C}_f}$.

We used here Dirac notation. Likewise, $\langle c|$ denotes the dual of $|c\rangle$, i.e. the linear form on $\mathcal{H}_{\mathcal{C}_f}$ such that for all $d \in \mathcal{C}_f$, $\langle c|\,(|d\rangle)$, which is noted $\langle c|d\rangle$, is equal to δ_{cd}. These notations may then be combined the other way around, $|c\rangle\langle c'|$ being the linear transformation of $\mathcal{H}_{\mathcal{C}_f}$ such that $|c\rangle\langle c'||d\rangle$ is, quite naturally, equal to $\langle c'|d\rangle|c\rangle$.

States on $\mathcal{H}_{\mathcal{C}_f}$ are nonnegative hermitian operators of trace 1. For instance, for each superposition of configurations $|\psi\rangle$, $|\psi\rangle\langle\psi|$ is a state, called in this case a *pure* state. Physically, states describe the actual state of matter; they bear all the information that can be measured in the system. The cells of our CA are in the pure state $|\psi\rangle\langle\psi|$ when what lies on them is, with certainty, the superposition $|\psi\rangle$. Actually, each state can be approximated by convex combinations of pure states. It means that the actual physical state of our CA at some moment can be described as a (possibly infinite) sum $\sum_i p_i|\psi_i\rangle\langle\psi_i|$, where the p_i's are positive, $\sum_i p_i = 1$ and the $|\psi_i\rangle$'s are pairwise orthogonal.

We will be manipulating isometries a lot. Unitary operators should be well-known, but isometries are probably somewhat less familiar, so let us write down their definition. A linear operator $G : \mathcal{H}_{\mathcal{C}_f} \longrightarrow \mathcal{H}_{\mathcal{C}_f}$ is *isometric* if and only if $\{G|c\rangle \mid c \in \mathcal{C}_f\}$ is an orthonormal family of $\mathcal{H}_{\mathcal{C}_f}$. This can also be expressed simply using the adjoint $G^\dagger$ of G. By definition, when G is a endomorphism of $\mathcal{H}_{\mathcal{C}_f}$, $G^\dagger$ is the endomorphism of $\mathcal{H}_{\mathcal{C}_f}$ such that for every $|\varphi\rangle, |\psi\rangle \in \mathcal{H}_{\mathcal{C}_f}$, $\langle\varphi|G|\psi\rangle = \langle\psi|G^\dagger|\varphi\rangle$. This way, $G^\dagger$ is indeed always unique; however, it is defined if and only if G is continuous. When G is isometric, G is of course continuous, and actually, G is isometric iff $G^\dagger G = \mathrm{Id}_{\mathcal{C}_f}$. If, moreover, G is onto, it is said to be *unitary*; so G is unitary if and only if $G^\dagger G = GG^\dagger = \mathrm{Id}_{\mathcal{C}_f}$.

Now, we are talking about CA, whose one important feature is shift-invariance. The definition of shift-invariance in a quantum context, with all these linearizations, is actually no more tedious than in the classical case; here it is.

Definition 1.2 (Shift-invariance)**.**
Consider the shift operation which takes configuration $\ldots c_{i-1}c_ic_{i+1}\ldots$ to $\ldots c'_{i-1}c'_ic'_{i+1}\ldots$ where, for all i, $c'_i = c_{i+1}$. Let $\sigma : \mathcal{H}_{\mathcal{C}_f} \longrightarrow \mathcal{H}_{\mathcal{C}_f}$ be its linear extension. A linear operator $G : \mathcal{H}_{\mathcal{C}_f} \longrightarrow \mathcal{H}_{\mathcal{C}_f}$ is said to be *shift invariant* if and only if $G\sigma = \sigma G$.

The second important feature of CA is their locality. In the classical case, we know that the locality is equivalent to the continuity of the global evolution on infinite configurations. Unfortunately, there does not seem to be such a result in the quantum case; at least it is not obvious what the right topology on superpositions of configurations should be. Therefore, the definition of locality proposed in [8] is more concrete. It explicitly states that to know the state of some region of the space after an iteration of the CA, you only need to know the state of a slightly larger region beforehand. In the classical case, you would trivially deduce from this property that the global evolution stems from a local transition rule. In the quantum case however, things are not so simple as entanglement suddenly comes into play, and when G is unitary it turns out [8, 2, 3] you need to keep things locally reversible.

To give the actual definition of locality, we first need to introduce some vocabulary. First, we will make abundant use throughout this paper of the Minkowski sum. For two subsets $\mathcal{A}$ and $\mathcal{B}$ of $\mathbb{Z}^d$, the Minkowski sum of $\mathcal{A}$ and $\mathcal{B}$, noted $\mathcal{A} + \mathcal{B}$, is the set $\{a + b/a \in \mathcal{A}, b \in \mathcal{B}\}$. $\mathcal{A} - \mathcal{B}$ is naturally the Minkowski difference, $\{a - b/a \in \mathcal{A}, b \in \mathcal{B}\}$.

$\mathcal{H}_{\mathcal{C}_f}$ has a natural structure of tensor product. Namely, for a subset $\mathcal{A}$ of $\mathbb{Z}^d$, let us note $\mathcal{C}_f(\mathcal{A})$ the set of the finite words on $\mathcal{A}$. Then $\mathcal{H}_{\mathcal{C}_f}$ is naturally isomorphic to $\mathcal{H}_{\mathcal{C}_f(\mathcal{A})} \otimes \mathcal{H}_{\mathcal{C}_f(\overline{\mathcal{A}})}$, where $\overline{\mathcal{A}}$ denotes the complementary of $\mathcal{A}$ in $\mathbb{Z}^d$ and $\mathcal{H}_{\mathcal{C}_f(\mathcal{A})}$ is the Hilbert space whose canonical basis is indexed by the elements of $\mathcal{C}_f(\mathcal{A})$. That being said, there are two more definitions we need before moving on. The first one should be familiar, it is also known as "trace out" and occurs whenever a quantum system can be divided into two subsystems. Informally, if a system S can be written as the tensor product of two subsystems A and B, and given a state ρ on S, you can chose to ignore completely what is going on B and restrict your universe to A. The state you get on A is then the restriction of ρ to A.

Definition 1.3 (Reduction)**.** Let ρ be a state over $\mathcal{H}_{\mathcal{C}_f}$ and $\mathcal{A}$ a subset of $\mathbb{Z}^d$. One can write $\rho = \sum_i \sigma_i \otimes \tau_i$, where the σ_i's and τ_i's are respectively operators over $\mathcal{H}_{\mathcal{C}_f(\mathcal{A})}$ and $\mathcal{H}_{\mathcal{C}_f(\overline{\mathcal{A}})}$. Then $\rho|_{\mathcal{A}}$, the *reduction* of ρ to $\mathcal{A}$, is a state on $\mathcal{H}_{\mathcal{C}_f(\mathcal{A})}$ defined as $\sum_i \mathrm{Tr}\,(\tau_i)\,\sigma_i$; this does not depend on the way ρ was decomposed in the first place. ■

The following definition is the dual of the last one. Why it is its dual will appear in proposition 1.6.

Definition 1.4 (Localization)**.** A linear endomorphism of $\mathcal{H}_{\mathcal{C}_f}$ is *localized* in a subset $\mathcal{A}$ of $\mathbb{Z}^d$ if it is of the form $A \otimes \mathrm{Id}$, where A is an endomorphism of $\mathcal{H}_{\mathcal{C}_f(\mathcal{A})}$ and Id is the identity on $\mathcal{H}_{\mathcal{C}_f(\overline{\mathcal{A}})}$.

We can now explain the duality going on here with this lemma, which is stated and proved as lemma 3 in [2].

Lemma 1.5 (Duality)**.**
Let $\mathcal{H}_0$ and $\mathcal{H}_1$ be Hilbert spaces, with $\mathcal{H}_0$ of finite dimension p. Let A, ρ, ρ' denote some elements of $\mathcal{L}(\mathcal{H}_0 \otimes \mathcal{H}_1)$ with ρ, ρ' having reductions $\rho|_0, \rho'|_0$ over $\mathcal{H}_0$. We then have that A is localized in $\mathcal{H}_0$ iff

"for every states ρ and ρ', if $\rho|_0 = \rho'|_0$ then $Tr(A\rho) = Tr(A\rho')$".

Moreover we have that $\rho|_0 = \rho'|_0$ is equivalent to

"if A is localized in $\mathcal{H}_0$, then $Tr(A\rho) = Tr(A\rho')$". ■

The proposition 1.6 we introduce next comes from theorem 3 in [2]. It entails structural reversibility, i.e. the fact that the inverse function of a RQCA is also a RQCA. Since we want now to talk about nonunitary operators, we have to restate it for general linear operators. We also have to extend the domain of localization of this operator from one cell to a set of cells. It will also serve as a definition of locality for linear endomorphisms over $\mathcal{H}_{\mathcal{C}_f}$ — which is not to be confused with localization. Note that the hypothesis of continuity for G provides the existence of its adjoint $G^\dagger$.

It defines the locality "at somewhere" in the space. Intuitively, a global transition is said to be local at some locus if the physical state in this locus after the transition depends only on the physical state on a neighbourhood of this locus beforehand (this is point (i) of the proposition). Equivalently, one could say that the result of each measure done on this locus after the transition could be predicted beforehand by measures performed on its neighbourhood (that would be point (ii)).

Proposition 1.6 (Structural reversibility)**.**
Let G be a continuous linear endomorphism of $\mathcal{H}_{\mathcal{C}_f}$, $\mathcal{A}$ and $\mathcal{N}$ respectively a subset and a finite subset of $\mathbb{Z}^d$. Suppose $G^\dagger$ The two properties are equivalent:

(i) *For every states ρ and ρ', if $\rho|_{\mathcal{A}+\mathcal{N}} = \rho'|_{\mathcal{A}+\mathcal{N}}$ then $\left(G\rho G^\dagger\right)|_{\mathcal{A}} = \left(G\rho' G^\dagger\right)|_{\mathcal{A}}$.*

(ii) *For every operator A localized in $\mathcal{A}$, $G^\dagger A G$ is localized in $\mathcal{A}+\mathcal{N}$.*

When G satisfies these properties, we say that G is local at $\mathcal{A}$ with neighbourhood $\mathcal{N}$. If only $\mathcal{A}$ is given, we say that G is local at $\mathcal{A}$ if there exists a finite subset $\mathcal{N}$ of $\mathbb{Z}^d$ such that G is local at $\mathcal{A}$ with neighbourhood $\mathcal{N}$.

If G is unitary, the following items are equivalent to (i) and (ii).

(iii) *For every states ρ and ρ' over the finite configurations, if $\rho|_{\mathcal{A}-\mathcal{N}} = \rho'|_{\mathcal{A}-\mathcal{N}}$ then $\left(G^\dagger \rho G\right)|_{\mathcal{A}} = \left(G^\dagger \rho' G\right)|_{\mathcal{A}}$.*

(iv) *For every operator A localized in $\mathcal{A}$, then $GAG^\dagger$ is localized on the cells in $\mathcal{A}-\mathcal{N}$.*

Proof.
$[(i) \Rightarrow (ii)]$. Suppose (i) and let A be an operator acting on cell 0. For every states ρ and ρ' such that $\rho|_{\mathcal{N}} = \rho'|_{\mathcal{N}}$, we have $\operatorname{Tr}\left(AG\rho G^\dagger\right) = \operatorname{Tr}\left(AG\rho' G^\dagger\right)$, using lemma 1.5 and our hypothesis that $\left(G\rho G^\dagger\right)|_{\mathcal{A}} = \left(G\rho' G^\dagger\right)|_{\mathcal{A}}$. We thus get $\operatorname{Tr}\left(G^\dagger AG\rho\right) = \operatorname{Tr}\left(G^\dagger AG\rho'\right)$. Since this is true of every ρ and ρ' such that $\rho|_{\mathcal{A}+\mathcal{N}} = \rho'|_{\mathcal{A}+\mathcal{N}}$, this means, again according to lemma 1.5, that $G^\dagger AG$ is localized on the cells in $\mathcal{A}+\mathcal{N}$.
$[(ii) \Rightarrow (i)]$. Suppose (ii) and $\rho|_{\mathcal{A}+\mathcal{N}} = \rho'|_{\mathcal{A}+\mathcal{N}}$. Then, for every operator B localized on the cells in $\mathcal{N}$, lemma 1.5 gives $\operatorname{Tr}(B\rho) = \operatorname{Tr}(B\rho')$, so for every operator A localized on cell 0, we get:

$$\begin{aligned}
\operatorname{Tr}\left(AG\rho G^\dagger\right) &= \operatorname{Tr}\left(G^\dagger AG\rho\right) \\
&= \operatorname{Tr}\left(G^\dagger AG\rho'\right) \\
\operatorname{Tr}\left(AG\rho G^\dagger\right) &= \operatorname{Tr}\left(AG\rho' G^\dagger\right)
\end{aligned}$$

Again by lemma 1.5, this means $\left(G\rho G^\dagger\right)|_0 = \left(G\rho' G^\dagger\right)|_0$.
Let us now assume G is unitary.
$[(ii) \Rightarrow (iv)]$. Suppose (ii) and let A be an operator acting on cell 0. Consider some operator M acting on a cell i which does not belong to $-\mathcal{N}$. According to our hypothesis we know that $G^\dagger MG$ does not act upon cell 0, and hence it commutes with A. But $AB \mapsto GAG^\dagger GBG^\dagger = GABG^\dagger$ is a morphism, hence $GG^\dagger MGG^\dagger = M$ also commutes with $GAG^\dagger$. Because M can be chosen amongst to full matrix algebra $M_d(\mathbb{C})$ of cell i, this entails that $GAG^\dagger$ must be the identity upon this cell. The same can be said of any cell outside $-\mathcal{N}$.
$[(iv) \Rightarrow (ii)]$, $[(iii) \Rightarrow (iv)]$, $[(iii) \Leftarrow (iv)]$ are symmetrical to $[(ii) \Rightarrow (iv)]$, $[(i) \Rightarrow (ii)]$, $[(ii) \Leftarrow (i)]$ just by interchanging the roles of G and $G^\dagger$. ∎

We can now say that again in a mathematically rigorous way: a RQCA is a unitary operator on $\mathcal{H}_{\mathcal{C}_f}$ that is shift-invariant and local at the central cell. Indeed, in this case, the assumption of locality at the central cell implies the locality at each finite subset $\mathcal{A}$ of $\mathbb{Z}^d$. Moreover, this locality is uniform, in the sense that a same neihbourghood $\mathcal{N}$ can be chosen for all $\mathcal{A}$'s. However, if we remove this hypothesis of unitarity, things are not so simple and we have to make stronger hypotheses; hence the following definitions.

Definition 1.7 (Locality). A continuous linear endomorphism G of $\mathcal{H}_{\mathcal{C}_f}$ is *everywhere local* if, for every finite subset $\mathcal{A}$ of $\mathbb{Z}^d$, G is local at $\mathcal{A}$. It is *uniformly local* if there exists a finite subset $\mathcal{N}$ of $\mathbb{Z}^d$ such that for every finite subset $\mathcal{A}$ of $\mathbb{Z}^d$, G is local at $\mathcal{A}$ with neighbourhood $\mathcal{N}$.

2. Linearization of Classical Automata

Let $F : \mathcal{C}_f \to \mathcal{C}_f$ be a cellular automaton on finite configurations, $\widetilde{F} : \mathcal{H}_{\mathcal{C}_f} \to \mathcal{H}_{\mathcal{C}_f}$ its linearization. For it to have a physical meaning and earn its name of "quantization", it should be an isometry, i.e. F should be one-to-one. We will nevertheless make a seemingly weaker assumption; we only assume $\widetilde{F}^\dagger$ to be defined, in order to be able to apply definition 1.7 and ask when $\widetilde{F}$ is local; it turns out that this condition actually implies the injectivity of F.

In order for $\widetilde{F}^\dagger$ to be defined, we have to assume that $\widetilde{F}$ is continuous. Beware that this notion of continuity has nothing to do with any kind of topology on the set of words, and is therefore not related to the continuity of F, which is true by definition of a CA. For $\widetilde{F}$ to be continuous means that it is bounded on the unit sphere of $\mathcal{H}_{\mathcal{C}_f}$. This is equivalent to saying that F is one-to-one. To verify this, let us first assume F is one-to-one. Then $\widetilde{F}^\dagger$ is isometric, and consequently continuous. Let us now assume F is not one-to-one. Since F is defined on the finite configurations, for every n, there exists $x_n \in \mathcal{C}_f$ such that x_n has a number of antecedents μ_n greater than n — just repeat as many times as needed some finite configuration having several antecedents. But then $\frac{1}{\sqrt{\mu_n}} \sum_{y\in\mathcal{C}_f/F(y)=x_n} |y\rangle$ is a unit vector whose image by $\widetilde{F}$, $\sqrt{\mu_n}|x_n\rangle$, has norm $\sqrt{\mu_n}$; hence, $\widetilde{F}$ is not bounded on the unit sphere, i.e. not continuous. To close this chapter on $\widetilde{F}^\dagger$, note that when it exists, it is defined as such:

$$\widetilde{F}^\dagger|a\rangle = \sum_{u\in\mathcal{C}_f/F(u)=a} |u\rangle.$$

We therefore assume from now on that F is one-to-one. So, if you are given a word w in the image of F, there is a unique $u \in \mathcal{C}_f$ such that $F(u) = w$. In general though, u can not be computed locally from w. If it were possible to do that, the cellular automaton would be, by definition, reversible, and thus, according to [8], its linearization would be a *bona fide* reversible quantum cellular automaton.

We will be monitoring XOR as an example, for which we will allow the quiescent state q to be renamed 0, the only letter in Σ being 1. XOR acts exactly as the usual XOR: it sums modulo 2 the bits in its neighbourhood $\{0; 1\}$. Of course, XOR is not reversible, since $11\ldots1$ and $00\ldots0$ are sent locally on the same word. It is one-to-one on finite configurations, though, while not surjective. It was already stated in proposition 1 of [2] that the quantization of nonreversible automata that are bijective on finite configurations could not be local, but that left the case of such automata as XOR unsettled. The following theorem does the job.

Theorem 2.1. *Suppose F is one-to-one. Then $\widetilde{F}$ is uniformly local if and only if F is reversible.*

Proof. Let us first briefly justify that when F is reversible, $\widetilde{F}$ is uniformly local. This is essentially what states the lemma 4 of [8], though in this case it is the automaton as defined on infinite configurations that is quantized. It is quite straightforward to adapt the statement and the proof of this lemma to our formalism, to get the same result: if F admits a neighbourhood $\mathcal{N}_C$ and an inverse neighbourhood $\mathcal{N}_I$, then $\mathcal{N}_C - \mathcal{N}_C + \mathcal{N}_I$ is a neighbourhood $\overline{\mathcal{N}}$; this is actually a direct consequence of lemma 3.2. However, there is a much simpler proof that such a neighbourhood exists. First, decompose your automaton into block permutations, with auxiliary bits if needed. Linearize then each of these block permutations. The composition of all these local unitary transformations is then $\widetilde{F} \otimes \mathrm{Id}$, where Id is the identity on the auxiliary qubits, and the block decomposition from which it is constructed is a witness that $\widetilde{F}$ is uniformly local.

We now prove the other implication, in a way that can be seen as a generalization of the argument presented page 7 of [2]. It proceeds by contraposition, so let us first of all assume F is not reversible. We will prove that for every set $\mathcal{N}$ there exists a set $\mathcal{A}$ such that $\widetilde{F}$ cannot satisfy the condition (i) of proposition 1.6; this will mean that $\widetilde{F}$ is not uniformly local.

Let $\mathcal{N}$ be a finite subset of $\mathbb{Z}^d$. Since F is not reversible, there exists a finite subset $\mathcal{B}$ of $\mathbb{Z}^d$ such that $F(x)|_{\mathcal{B}-\mathcal{N}} = F(y)|_{\mathcal{B}-\mathcal{N}}$ but $x|_{\mathcal{B}} \neq y|_{\mathcal{B}}$. Let $\mathcal{A} = \left\{s \in \mathbb{Z}^d / F(x)|_s \neq F(y)|_s\right\}$; since $F(x)$ and $F(y)$ both are finite configurations, $\mathcal{A}$ is finite.

Let $|\varphi_\pm\rangle$ denote the superpositions of configurations $\frac{|x\rangle \pm |y\rangle}{\sqrt{2}}$, and let $\rho_\pm$ be the pure states $|\varphi_\pm\rangle\langle\varphi_\pm|$. We are now going to prove that $\rho_+|_{\mathcal{A}+\mathcal{N}} = \rho_-|_{\mathcal{A}+\mathcal{N}}$, while $\left(\widetilde{F}\rho_+\widetilde{F}^\dagger\right)\Big|_{\mathcal{A}} \neq \left(\widetilde{F}\rho_-\widetilde{F}^\dagger\right)\Big|_{\mathcal{A}}$.

Since $F(x)$ and $F(y)$ are equal on $\mathcal{B}-\mathcal{N}$, $\mathcal{A}+\mathcal{N}$ does not intersect $\mathcal{B}$, so x and y differ on some point on the complement of $\mathcal{A}+\mathcal{N}$. Considering the partition of $\mathbb{Z}^d$ into $\mathcal{A}+\mathcal{N}$ and $\overline{\mathcal{A}+\mathcal{N}}$, we can thus write $|x\rangle = |x_1\rangle \otimes |x_2\rangle$ and $|y\rangle = |y_1\rangle \otimes |y_2\rangle$, where $x_1, y_1 \in \mathcal{C}_f\left(\mathcal{A}+\mathcal{N}\right)$, $x_2, y_2 \in \mathcal{C}_f\left(\overline{\mathcal{A}+\mathcal{N}}\right)$, and $x_2 \neq y_2$. We then have

$$\begin{aligned}
\rho_\pm|_{\mathcal{A}+\mathcal{N}} &= \frac{1}{2}\left(|x\rangle\langle x| \pm |x\rangle\langle y| \pm |y\rangle\langle x| + |y\rangle\langle y|\right)|_{\mathcal{A}+\mathcal{N}} \\
&= \frac{1}{2}\left.\left(\begin{array}{c} |x_1\rangle\langle x_1| \otimes |x_2\rangle\langle x_2| \pm |x_1\rangle\langle y_1| \otimes |x_2\rangle\langle y_2| \\ \pm|y_1\rangle\langle x_1| \otimes |y_2\rangle\langle x_2| + |y_1\rangle\langle y_1| \otimes |y_2\rangle\langle y_2| \end{array}\right)\right|_{\mathcal{A}+\mathcal{N}} \\
\rho_\pm|_{\mathcal{A}+\mathcal{N}} &= \frac{1}{2}\left(|x_1\rangle\langle x_1| + |y_1\rangle\langle y_1|\right).
\end{aligned}$$

Thus, the reductions of ρ_+ and ρ_- and $\mathcal{A}+\mathcal{N}$ are indeed equal. Now, $\widetilde{F}\rho_\pm\widetilde{F}^\dagger = |\psi_\pm\rangle\langle\psi_\pm|$, where $|\psi_\pm\rangle = \frac{|F(x)\rangle \pm |F(y)\rangle}{\sqrt{2}}$. Since $F(x)$ and $F(y)$ coincide on $\overline{\mathcal{A}}$, we actually have $\widetilde{F}\rho_\pm\widetilde{F}^\dagger = \sigma_1 \otimes \sigma_\pm$, where σ_1 is a (pure) state over $\mathcal{H}_{\mathcal{C}_f(\overline{\mathcal{A}})}$, and the $\sigma_\pm$'s are states over $\mathcal{H}_{\mathcal{C}_f(\mathcal{A})}$. The reductions of $\widetilde{F}\rho_\pm\widetilde{F}^\dagger$ to $\mathcal{A}$ are then $\sigma_\pm$, which are distinct states since ρ_+ and ρ_- where distinct to begin with. ∎

Another way to present this proof is to appeal to the perennial Alice and Bob. We start with the state ρ_+. Alice and Bob have access to some cells of $\mathbb{Z}^d$, meaning that they can conjugate the state on $\mathcal{H}_{\mathcal{C}_f}$ with unitary operators, as long as these unitary operators are localized in the region of the space they were assigned. So let Alice and Bob's regions be respectively $\mathcal{A}$ and $\mathcal{B}$ as encountered in the proof of theorem 2.1. We will see how they can communicate through the use of $\widetilde{F}$, even though their regions could be at quite a large distance from each other, depending on $\mathcal{N}$.

Since $x|_{\mathcal{B}} \neq y|_{\mathcal{B}}$, Bob is able to transform at will ρ_+ into ρ_-, by performing a *controlled phase-shift* on some cell where x and y differ. What that means informally is that, since Bob is able to tell the difference between x and y in his area, he can introduce a dissimetry between $|x\rangle$ and $|y\rangle$. Of course he could simply transform $|y\rangle$ by changing the letters of y is some cells or something like that, but that would not allow him to communicate any faster than in the classical case. So what Bob does is to change $|y\rangle$ into $-|y\rangle$, something more immaterial, purely quantum and, in a way "delocalized", that will allow Alice to catch his message, which is one bit of information : "did I or didn't I change ρ_+ into ρ_-?". After Bob did his thing, $\widetilde{F}$ is applied to the state.

Now, Alice being able to actually read the message is due to the fact that her region contains all the cells where $F(x)$ and $F(y)$. As explained in the proof of theorem 2.1, the state after $\widetilde{F}$ has been applied is a tensor product of a state on $\mathcal{A}$ and a state on $\overline{\mathcal{A}}$, the state on $\overline{\mathcal{A}}$ not depending on the prior actions of Bob; therefore, the state on $\mathcal{A}$ does depend on them, so Alice must have a way to distinguish between them — in this case she just has to perform a so-called *swap-test*. Let us see for instance what happens with XOR. Consider these two words in $\mathcal{C}_f$:

$$\begin{array}{rcl} x & = & \ldots 0000000000000000\ldots \\ y & = & \ldots 0011111111111100\ldots \end{array}$$

Their images are

$$\begin{array}{rcl} F(x) & = & \ldots 0000000000000000\ldots \\ F(y) & = & \ldots 0100000000000100\ldots \end{array}$$

Now put Bob on the middle of the stripe, and Alice at the two cells where the 1's are in $F(y)$. By following the protocol described in the proof of theorem 2.1, Bob can indeed send a bit of information to Alice. There is no doubt that this is a correct proof that $\widetilde{\text{XOR}}$ is not uniformly local, but one might argue that this idea of an "Alice" surrounding Bob makes little sense: surely if Alice can be present at two faraway places in the stripe at the same time, it means he must have some way to go from one place to the other, and since in the middle stands Bob, why would she bother using $\widetilde{\text{XOR}}$ to send her message? Cannot we find another protocol where Alice stands either on the left or on the right of Bob, but on only one side at a time? Actually, no, we cannot, and this is related to the fact that $\widetilde{\text{XOR}}$, while not uniformly local, is still everywhere local: if Bob is forbidden the access to the cells located between Alice's positions, then he cannot transmit her any message. The proof of this assertion is the object of the next section.

3. Everywhere Locality in the One-dimensional Case

The question is: when is the quantization of a one-to-one CA everywhere local? We are going now to give a proof that in the one-dimensional case, it is equivalent to the openness of F_∞, the extension of F to the set $\mathcal{C}_\infty$ of infinite configurations; so let us fix the dimension d to 1 for this section.

First, it might be useful to remind what it means for F_∞ to be open. $\mathcal{C}_\infty$ comes with the usual topology; namely, a base of open sets is given by the sets $\{v \in \mathcal{C}_\infty / v_{\mathcal{A}} = w_{\mathcal{A}}\}$, for $w \in \mathcal{C}_\infty$ and $\mathcal{A}$ a finite subset of Z^d. By definition, F_∞ is *open* if for every open subset O of $\mathcal{C}_\infty$, $F_\infty(O)$ is open.

Proposition 3.1. *$\widetilde{F}$ is everywhere local if and only if F_∞ is open.*

Proof. We will appeal to [7]. According to its theorem 5.45, F_∞ is open iff it is left and right-closing. The definitions of left and right-closingness may be found in definition 5.38. First, x and y in $\mathcal{C}_\infty$ are said to be *left-asymptotic* (respectively *right-asymptotic*) when there is some $n \in \mathbb{Z}$ such that for every $k < n$ (resp. $k > n$), $x_k = y_k$. By definition, F_∞ is left-closing (respectively right-closing) if, for every $x, y \in \mathcal{C}_\infty$ that are left-asymptotic (resp. right-asymptotic), if $F(x) = F(y)$ then $x = y$. We now translate these conditions on de Bruijn diagrams.

Let us recall briefly what we mean by Bruijn diagrams. Let n be an integer such that $[-n; n+1]$ is a neighbourhood for F. We note F_0 the function from $(q\Sigma)^{[-n;n+1]}$ to $q\Sigma$ which computes locally F on cell 0, from the knowledge of the stripe on $[-n; n+1]$. Then the associated de Bruijn diagram is a graph whose vertices are indexed by the pairs $(u, v) \in q\Sigma^{[-n;n]} \times q\Sigma^{[-n;n]}$. There is an edge from (u, v) to (u', v') if and only if

- for $i \in [-n; n[$, $u_{i+1} = u'_i$ and $v_{i+1} = v'_i$
- $F_0(u_{-n}u_{-n+1} \ldots u_n u'_n) = F_0(v_{-n}v_{-n+1} \ldots v_n v'_n)$.

The first thing we want to note is that the strongly connected component (SCC) of (q, q) in the de Bruijn diagram includes the diagonal Δ of $q\Sigma^{[-n;n]} \times q\Sigma^{[-n;n]}$, i.e. the elements of the form (u, u).

To each pair of words $(u, v) \in \mathcal{C}_\infty \times \mathcal{C}_\infty$ such that $F(u) = F(v)$ is associated a bi-infinite path on the de Bruijn diagram, and vice-versa. In this respect, we see that "F_∞ is left-closing" is equivalent to "every infinite path starting from Δ stays forever in Δ", while "F_∞ is left-closing" is the dual statement that "every bi-infinite path ending in Δ is completely included in Δ". Thus, F_∞ is open iff there is no connection, in or out, between Δ and any cycle of the de Brujin diagram not included in Δ.

Now, what does it mean on this diagram for $\widetilde{F}$ to be everywhere local? If we follow the proof of theorem 2.1, we see this means that there exists an integer k such that for every integer n, if $F(x)$ is known on $\overline{[-n; n]}$, then $x \in \mathcal{C}_f$ is determined on $\overline{[-n-k; n+k]}$. On the de Bruijn diagram, it means that there exists an integer k such that any path starting from (q, q) must stay in X until k steps before the end, and that every path ending in (q, q) must stay in X after k steps. This also means that X is not connected to any cycle not included in Δ.

Suppose F_∞ is not open. Without loss of generality, we assume there is a path from a cycle not included in Δ to Δ. This cycle is given by two distinct finite words v and v' of same length such that $F(\ldots vvvv \ldots) = F(\ldots v'v'v'v' \ldots)$; the path from this cycle to (q, q) is given by two words of same lenght w and w', such that $F(\ldots vvvwqqq \ldots) = F(\ldots v'v'v'w'qqq \ldots)$. Let $[-n; n]$ be a neighbourhood for F and k a positive integer. Now consider the finite configurations $x_k = \ldots qqqv^k wqqq \ldots$ and $y_k = \ldots qqqv'^k w'qqq \ldots$, where the first letter of the first v has position 0. Almost everywhere, (x_k, y_k) follows a path on the de Bruijn diagram. The only points where (x_k, y_k) does not follow an edge of this diagram is at the transition between cells -1 and 0. So $\mathcal{A}_k = \{i \in \mathbb{Z}/F(x_k) \neq F(y_k)\}$ is included in $[-n-1; n]$, and does not depend on k when k is large enough; let's define $\mathcal{A} = \lim_{k\to\infty} \mathcal{A}_k$. Let $\mathcal{B}_k \subseteq \mathbb{Z}$ be the singleton consisting of the rightmost cell where x_k and y_k differ. Since $v \neq v'$, its emplacement is at least $k - 1$. Let $\mathcal{N}$ be a finite subset of $\mathbb{Z}$; for a large enough k, we have the following properties:

- $F(x_k)|_{\mathcal{B}_k - \mathcal{N}} = F(y_k)|_{\mathcal{B}_k - \mathcal{N}}$
- $x_k|_{\mathcal{B}_k} \neq y_k|_{\mathcal{B}_k}$
- $\mathcal{A} = \{i \in \mathbb{Z}/F(x_k) \neq F(y_k)\}$.

Then, according to the proof of theorem 2.1, $\widetilde{F}$ is not local at $\mathcal{A}$ with neigbourhood $\mathcal{N}$. Since we showed that there exists $\mathcal{A}$ such that this is true for any $\mathcal{N}$, we have indeed just proven that $\widetilde{F}$ is not everywhere local.

Now, what remains to prove is that when F_∞ is open, $\widetilde{F}$ is everywhere local. To do that we will strengthen a little bit the lemma 4 of [8]. But first we need to explain a property of one-dimensional open automata. Suppose F_∞ is open and let $\mathcal{A}$ be a finite subset of $\mathbb{Z}$ and x and y two words such that $F(x)|_{\overline{\mathcal{A}}} = F(y)|_{\overline{\mathcal{A}}}$. Say $\mathcal{A}$ is included in $[-n;n]$, $[-k;k]$ is a neighbourhood for F and l is the number of vertices in the de Bruijn diagram. If we look at (x,y) as a run in this diagram, then we follow edges except perhaps in $[-n-k;n+k]$. But since there are no loops connected in one way or another do Δ, and we have to join Δ at $\pm\infty$, this means we are always in Δ except perhaps in $[-n-k-l;n+k+l]$, to give a rough bound. So there exists a finite subset $\mathcal{N}_I$ of $\mathbb{Z}$, which does not depend on x nor y — though it may depend on $\mathcal{A}$ — such that $x|_{\overline{\mathcal{A}+\mathcal{N}_I}} = y|_{\overline{\mathcal{A}+\mathcal{N}_I}}$. Now all is needed to complete the proof is the next (and last) lemma, which, as announced, is but a gentle strengthening of the lemma 4 of [8].

Lemma 3.2. *Let F be a one-to-one automaton with neighbourhood $\mathcal{N}_C$. Let $\mathcal{A}$ and $\mathcal{N}_I$ be finite subsets of $\mathbb{Z}$ such that for all $x,y \in \mathcal{C}_f(\mathbb{Z})$, if $F(x)|_{\overline{\mathcal{A}}} = F(y)|_{\overline{\mathcal{A}}}$, then $x|_{\overline{\mathcal{A}+\mathcal{N}_I}} = y|_{\overline{\mathcal{A}+\mathcal{N}_I}}$. Suppose $\mathcal{N}_C$ and $\mathcal{N}_I$ contain 0. Then $\widetilde{F}$ is local at $\mathcal{A}$ with neighbourhood $\mathcal{N} = \mathcal{N}_C - \mathcal{N}_C + \mathcal{N}_I$.*

Proof. Let $\mathcal{A} \subseteq \mathbb{Z}^d$. Let ρ and ρ' be states over $\mathcal{H}_{\mathcal{C}_f}$ such that $\rho|_{\mathcal{A}+\mathcal{N}} = \rho'|_{\mathcal{A}+\mathcal{N}}$. We have to prove $\left(\widetilde{F}\rho\widetilde{F}^\dagger\right)|_{\mathcal{A}} = \left(\widetilde{F}\rho'\widetilde{F}^\dagger\right)|_{\mathcal{A}}$.

Let us write $\rho = \sum_{a,b\in\mathcal{C}_f} \lambda_{a,b}|a\rangle\langle b|$ and $\rho' = \sum_{a,b\in\mathcal{C}_f} \lambda'_{a,b}|a\rangle\langle b|$. Then

$$\rho|_{\mathcal{A}+\mathcal{N}} = \sum_{a,b/a_{\overline{\mathcal{A}+\mathcal{N}}}=b_{\overline{\mathcal{A}+\mathcal{N}}}} \lambda_{a,b}|a_{\mathcal{A}+\mathcal{N}}\rangle\langle b_{\mathcal{A}+\mathcal{N}}| = \sum_{x,y\in A^{\mathcal{A}+\mathcal{N}}} \left(\sum_{u\in A^{\overline{\mathcal{A}+\mathcal{N}}}} \lambda_{x.u,y.u}\right)|x\rangle\langle y|.$$

Ergo, the hypothesis $\rho|_{\mathcal{A}+\mathcal{N}} = \rho'|_{\mathcal{A}+\mathcal{N}}$ may be translated as

$$\forall x,y \in A^{\mathcal{A}+\mathcal{N}} \quad \sum_{u\in A^{\overline{\mathcal{A}+\mathcal{N}}}} \lambda_{x.u,y.u} = \sum_{u\in A^{\overline{\mathcal{A}+\mathcal{N}}}} \lambda'_{x.u,y.u}.$$

For $x,y \in A^{\mathcal{A}+\mathcal{N}}$, let $\alpha(x,y)$ be the set of couples (a,b) of words in $\mathcal{C}_f$ such that $a_{\mathcal{A}+\mathcal{N}} = x$, $b_{\mathcal{A}+\mathcal{N}} = y$ and $a_{\overline{\mathcal{A}+\mathcal{N}}} = b_{\overline{\mathcal{A}+\mathcal{N}}}$. Then the hypothesis is equivalent to

$$\forall x,y \in A^{\mathcal{A}+\mathcal{N}} \quad \sum_{(a,b)\in\alpha(x,y)} \lambda_{a,b} = \sum_{(a,b)\in\alpha(x,y)} \lambda'_{a,b}. \tag{3.1}$$

Let us now try translating our aim in the same way. First we have

$$\widetilde{F}\rho\widetilde{F}^\dagger = \sum_{a,b\in\mathcal{C}_f} \lambda_{a,b}|F(a)\rangle\langle F(b)| = \sum_{c,d\in F(\mathcal{C}_f)} \lambda_{F^{-1}(c),F^{-1}(d)}|c\rangle\langle d|$$

$$\left(\widetilde{F}\rho\widetilde{F}^\dagger\right)|_{\mathcal{A}} = \sum_{c,d\in F(\mathcal{C}_f)/c_{\overline{\mathcal{A}}}=d_{\overline{\mathcal{A}}}} \lambda_{F^{-1}(c),F^{-1}(d)}|c_{\mathcal{A}}\rangle\langle d_{\mathcal{A}}|$$

$$\left(\widetilde{F}\rho\widetilde{F}^\dagger\right)|_{\mathcal{A}} = \sum_{z,t\in A^{\mathcal{A}}} \left(\sum_{u\in A^{\overline{\mathcal{A}}}} \lambda_{F^{-1}(z.w),F^{-1}(t.w)}\right)|z\rangle\langle t|.$$

So what we want to prove is that, for every z and t in $A^{\mathcal{A}}$,

$$\sum_{w\in A^{\overline{\mathcal{A}}}} \lambda_{F^{-1}(z.w),F^{-1}(t.w)} = \sum_{w\in A^{\overline{\mathcal{A}}}} \lambda'_{F^{-1}(z.w),F^{-1}(t.w)},$$

with the convention that these numbers are 0 when F^{-1} is not appliable. For $z,t \in A^{\mathcal{A}}$, let $\beta(z,t)$ be the set of couples (a,b) of words in $\mathcal{C}_f$ such that $F(a)_{\mathcal{A}} = z$, $F(b)_{\mathcal{A}} = t$ and $F(a)_{\overline{\mathcal{A}}} = F(b)_{\overline{\mathcal{A}}}$. What we want to prove from (3.1) is the equivalent to

$$\forall z,t \in A^{\mathcal{A}} \sum_{(a,b)\in\beta(z,t)} \lambda_{a,b} = \sum_{(a,b)\in\beta(z,t)} \lambda'_{a,b}. \tag{3.2}$$

We will prove this by showing that for each $z,t \in A^{\mathcal{A}}$, there is some set $\gamma(z,t)$ such that $\beta(z,t) = \coprod_{(x,y)\in\gamma(z,t)} \alpha(x,y)$, ie $\beta(z,t)$ is the disjoint union of the $\alpha(x,y)$'s for (x,y) in $\gamma(z,t)$.

On the one hand, it is quite immediate by definition that, when $(x,y) \neq (x',y')$, $\alpha(x,y)$ and $\alpha(x',y')$ are disjoint. On the other hand, by hypothesis, every (a,b) of $\beta(z,t)$ is in some $\alpha(x,y)$, so that $\gamma(z,t)$ may be found in this simple way: for each (a,b) in $\beta(z,t)$, find the unique $(x_{a,b},y_{a,b})$ such that (a,b) is in $\alpha(x_{a,b},y_{a,b})$, and then define $\gamma(z,t)$ to be the set of all these (x,y)'s you found. The only problem is that you could add unwanted (a,b)'s by doing so; we need only checking that this is not the case. In other words, we have to prove that whenever the intersection between $\alpha(x,y)$ and $\beta(z,t)$ is nonempty, then the former is included in the latter.

So, let (a,b) be an en element of $\alpha(x,y)\cap\beta(z,t)$ and (a',b') an other element of $\alpha(x,y)$. First of all, since a and a' coincide on $\mathcal{A}+\mathcal{N}$ (where they are equal to x), and in particular on $\mathcal{A}+\mathcal{N}_C$, then $f(a)$ and $f(a')$ coincide on $\mathcal{A}$, thus $F(a')_{\mathcal{A}} = F(a)_{\mathcal{A}} = z$. Likewise, of course, $F(b')_{\mathcal{A}} = t$.

Then, by hypothesis and since $\mathcal{A}$ is finite and $F(a)_{\overline{\mathcal{A}}} = F(b)_{\overline{\mathcal{A}}}$, a and b coincide on $\overline{\mathcal{A}+\mathcal{N}_I}$, not only on $\overline{\mathcal{A}+\mathcal{N}}$. This implies that x and y must coincide on $(\mathcal{A}+\mathcal{N})\cap\overline{\mathcal{A}+\mathcal{N}_I}$, and as a consequence a' and b' do also coincide on $\overline{\mathcal{A}+\mathcal{N}_I}$; thus $F(a')$ and $F(b')$ coincide on $\overline{\mathcal{A}+\mathcal{N}_I-\mathcal{N}_C}$.

Lastly, since a and a' coincide on $\mathcal{A}+\mathcal{N} = \mathcal{A}+\mathcal{N}_C-\mathcal{N}_C+\mathcal{N}_I$, so do $F(a)$ and $F(a')$ on $\mathcal{A}-\mathcal{N}_C+\mathcal{N}_I$. Likewise, $F(b)$ and $F(b')$ coincide on that same interval. However, $F(a)$ and $F(b)$ coincide on $\overline{\mathcal{A}}$, by hypothesis; ergo, $F(a')$ and $F(b')$ coincide on $\overline{\mathcal{A}}\cap(\mathcal{A}-\mathcal{N}_C+\mathcal{N}_I)$. Put it together, you finally get that $F(a')$ and $F(b')$ coincide on $\overline{\mathcal{A}}$; Q.E.D. ■

XOR$_\infty$ being, as can be checked easily on its de Bruijn diagram, open, it is thus everywhere local, which also means Alice has to surround Bob in order to receive his long-distance calls. On the contrary, the modified version of XOR that was defined in the definition 11 of [2] is not open on the infinite configurations, which is why we were able to find a protocol where Bob and Alice lie on two distinct sides of the stripe.

4. Conclusion

Starting only with the assumption that we should be able to use the adjoint of $\widetilde{F}$, this implied it should be isometric, thus convey a physical meaning as a valid quantum evolution. If we then add the constraint that it should be uniformly local — something that you would certainly expect a cellular automaton to verify in any model — it turns out F has to be reversible, so that $\widetilde{F}$ is part of the already well-known class of RQCA. This is good news in a way: the notion of a RQCA is a robust one; however, it could nevertheless be considered a downside. Indeed, as stated in the introduction, RQCA are now believed to be fairly well understood, so the next challenge is understanding nonreversible quantum cellular automata. It would certainly have been of great help to be able to construct such NRQCA by quantizing nonreversible CA. Alas, this paper shows that such a thing is impossible. Quantizing one-dimensional open non-reversible automata certainly provides puzzling entities, but no quantum CA; there remains however an interesting open question about the generalization of proposition 3.1 to higher dimensions.

Then again, the most important question right now is: what are NRQCA? Can they be defined from their global evolution in a reasonably simple way? This question, in its most general form, includes the same one concerning randomized automata instead of quantum ones, since classical randomness is part of the quantum world, and as far as we know this question has been little studied. Let us ask it in a more precise way: what is the property on the global evolution of probability distributions that characterizes randomized cellular automata, i.e. those transformations that can be written as a finite number of layers, each of them consisting of a tiling of identical blocks performing some local random transformation?

Acknowledgements

We would like to thank Guillaume Theyssier for pointing out useful references and Reinhard Werner for asking the right questions and providing useful advices and encouragement. Also, a special thanks to the reviewer who remarked that a proof was absent and another one unclear, two remarks leading to the discovery of two mistakes in the first version of the paper, which led in turn to substantial rewriting. Our best hope is not to have introduced too many new mistakes in the process.

References

[1] P. Arrighi, *An alegraic study of unitary one-dimensional quantum cellular automata*, Proceedings of MFCS 2006, LNCS 4162 (2006), 122–133, arXiv:quant-ph/0512040v2

[2] P. Arrighi, V. Nesme, R. Werner, *One-dimensional quantum cellular automata over finite, unbounded configurations*, arXiv:0711.3517v1.

[3] P. Arrighi, V. Nesme, R. Werner, *N-dimensional Quantum Cellular Automata*, arXiv:0711.3975v1

[4] C. Dürr, H. LêThanh, M. Santha, *A decision procedure for well formed quantum cellular automata*, Random Structures and Algorithms, **11**, 381–394, (1997).

[5] C. Dürr, M. Santha, *A decision procedure for unitary quantum linear cellular automata*, SIAM J. of Computing, **31**(4), 1076–1089, (2002).

[6] R. P. Feynman, *Quantum mechanical computers*, Found. Phys. **16**, 507–531, (1986).

[7] P. Kůrka, *Topological dynamics of cellular automata, Codes, Systems and Graphical Models* (B. Marcus and J. Rosenthal, eds.), The IMA Volumes in Mathematics and its Applications, 123, 447–386, Springer-Verlag, Berlin 2001.

[8] B. Schumacher, R. F. Werner, *Reversible quantum cellular automata*, arXiv:quant-ph/0405174.

[9] J. Watrous, *On one-dimensional quantum cellular automata*, Complex Systems **5**(1), 19–30, (1991).

Journées Automates Cellulaires 2008 (Uzès), pp. 216-227

TOPOLOGICAL PROPERTIES OF SAND AUTOMATA AS CELLULAR AUTOMATA

ALBERTO DENNUNZIO [1], PIERRE GUILLON [2], AND BENOÎT MASSON [3]

[1] Università degli studi di Milano-Bicocca, Dipartimento di Informatica Sistemistica e Comunicazione, via viale Sarca 336, 20126 Milano (Italy)
E-mail address: `dennunzio@disco.unimib.it`

[2] Université Paris-Est, Laboratoire d'Informatique de l'Institut Gaspard Monge, UMR CNRS 8049, 5 bd Descartes, 77454 Marne la Vallée Cedex 2 (France)
E-mail address: `pierre.guillon@univ-mlv.fr`

[3] Laboratoire d'Informatique Fondamentale de Marseille (LIF)-CNRS, Aix-Marseille Université, 39 rue Joliot-Curie, 13453 Marseille Cedex 13 (France)
E-mail address: `benoit.masson@lif.univ-mrs.fr`

Abstract. In this paper, we exhibit a strong relation between the sand automata configuration space and the cellular automata configuration space. This relation induces a compact topology for sand automata, and a new context in which sand automata are homeomorphic to cellular automata acting on a specific subshift. We show that the existing topological results for sand automata, including the Hedlund-like representation theorem, still hold. In this context, we give a characterization of the cellular automata which are sand automata.

1. Introduction

Self-organized criticality (SOC) is a common phenomenon observed in a huge variety of processes in physics, biology and computer science. A SOC system evolves to a "critical state" after some finite transient. Any perturbation, no matter how small, of the critical state generates a deep reorganization of the whole system. Then, after some other finite transient, the system reaches a new critical state and so on. Examples of SOC systems are: sandpiles, snow avalanches, star clusters in the outer space, earthquakes, forest fires, load balance in operating systems [2, 3, 16]. Among them, sandpiles models are a paradigmatic formal model for SOC systems [8, 9].

In [4], the authors introduced sand automata as a generalization of sandpiles models and transposed them in the setting of discrete dynamical systems. A key-point of [4] was

2000 ACM Subject Classification: F.1.1 Models of Computation.
Key words and phrases: sand automata, cellular automata, dynamical systems, subshifts.
This work has been supported by the Interlink/MIUR project "Cellular Automata: Topological Properties, Chaos and Associated Formal Languages", by the ANR Blanc "Projet Sycomore" and by the PRIN/MIUR project "Formal Languages and Automata: Mathematical and Applicative Aspects".

to introduce a (locally compact) metric topology to study the dynamical behavior of sand automata. A first and important result was a fundamental representation theorem similar to the well-known theorem for cellular automata from Hedlund [10, 4]. In [5, 6], the authors investigate sand automata by dealing with some basic set properties and decidability issues.

In this paper we continue the study of sand automata. First of all, we introduce a different metric on configurations (i.e., spatial distributions of sand grains). This metric is defined by means of the relation between sand automata and cellular automata [6]. With the induced topology, the configuration set turns out to be a compact (and not only locally compact), perfect and totally disconnected space. The "strict" compactness gives a better topological background to study the behavior of sand automata (and in general of discrete dynamical systems) [1, 12]. We show that all the topological results from [4], in particular the Hedlund-like representation theorem, remain valid with the compact topology. Moreover, with this topology, any sand automaton is homeomorphic to a cellular automaton defined on a subset of its usual domain. We prove that it is possible to decide whether a given cellular automaton represents, through that homeomorphism, a sand automaton.

The paper is structured as follows. In Section 2, we recall basic definitions and results about cellular automata and sand automata. Then, in Section 3, we define the topology and prove topological results, in particular the representation theorem.

2. Definitions

For all $a, b \in \mathbb{Z}$ with $a \leq b$, let $[a, b] = \{a, a+1, \ldots, b\}$ and $\widetilde{[a, b]} = [a, b] \cup \{+\infty, -\infty\}$. For $a \in \mathbb{Z}$, let $[a, +\infty) = \{a, a+1, \ldots\} \setminus \{+\infty\}$. Let $\mathbb{N}_+$ be the set of positive integers.

Let A a (possibly infinite) alphabet and $d \in \mathbb{N}^*$. Denote by $\mathcal{M}^d$ the set of all the d-dimensional matrices with values in A. We assume that the entries of any matrix $U \in \mathcal{M}^d$ are all the integer vectors of a suitable d-dimensional hyper-rectangle $[1, h_1] \times \cdots \times [1, h_d] \subset \mathbb{N}_+^d$. For any $h = (h_1, \ldots, h_d) \in \mathbb{N}_+^d$, let $\mathcal{M}_h^d \subset \mathcal{M}^d$ be the set of all the matrices with entries in $[1, h_1] \times \cdots \times [1, h_d]$. In the sequel, the vector h will be called the *order* of the matrices belonging to $\mathcal{M}_h^d$. For a given element $x \in A^{\mathbb{Z}^d}$, the *finite portion* of x of reference position $i \in \mathbb{Z}^d$ and order $h \in \mathbb{N}_+^d$ is the matrix $M_h^i(x) \in \mathcal{M}_h^d$ defined as $\forall k \in [1, h_1] \times \cdots \times [1, h_d]$, $M_h^i(x)_k = x_{i+k-\mathbf{1}}$. For any $r \in \mathbb{N}$, let $\mathbf{r}^d$ (or simply $\mathbf{r}$ if the dimension is not ambiguous) be the vector $(r, \ldots, r)$.

2.1. Cellular automata and subshifts

Let A be a finite alphabet. A *CA configuration* of dimension d is a function from $\mathbb{Z}^d$ to A. The set $A^{\mathbb{Z}^d}$ of all the CA configurations is called the *CA configuration space*. This space is usually equipped with the Tychonoff metric d_T defined by

$$\forall x, y \in A^{\mathbb{Z}^d}, \quad \mathsf{d}_T(x, y) = 2^{-k} \quad \text{where} \quad k = \min\left\{|j| : j \in \mathbb{Z}^d, x_j \neq y_j\right\} .$$

The topology induced by d_T coincides with the product topology induced by the discrete topology on A. With this topology, the CA configuration space is a Cantor space: it is compact, perfect (i.e., it has no isolated points) and totally disconnected.

For any $k \in \mathbb{Z}^d$ the *shift map* $\sigma^k : A^{\mathbb{Z}^d} \to A^{\mathbb{Z}^d}$ is defined by $\forall x \in A^{\mathbb{Z}^d}, \forall i \in \mathbb{Z}^d$, $\sigma^k(x)_i = x_{i+k}$. A function $F : A^{\mathbb{Z}^d} \to A^{\mathbb{Z}^d}$ is said to be *shift-commuting* if $\forall k \in \mathbb{Z}^d$, $F \circ \sigma^k = \sigma^k \circ F$.

A d-dimensional *subshift* S is a closed subset of the CA configuration space $A^{\mathbb{Z}^d}$ which is shift-invariant, i.e., for any $k \in \mathbb{Z}^d$, $\sigma^k(S) \subset S$. Let $\mathcal{F} \subseteq \mathcal{M}^d$ and let $S_\mathcal{F}$ be the set of configurations $x \in A^{\mathbb{Z}^d}$ such that all possible finite portions of x do not belong to $\mathcal{F}$, i.e., for any $i, h \in \mathbb{Z}^d$, $M_h^i(x) \notin \mathcal{F}$. The set $S_\mathcal{F}$ is a subshift, and $\mathcal{F}$ is called its set of forbidden patterns. Note that for any subshift S, it is possible to find a set of forbidden patterns $\mathcal{F}$ such that $S = S_\mathcal{F}$. A subshift S is said to be a *subshift of finite type* (SFT) if $S = S_\mathcal{F}$ for some finite set $\mathcal{F}$. The language of a subshift S is $\mathcal{L}(S) = \left\{U \in \mathcal{M}^d : \exists i \in \mathbb{Z}^d, h \in \mathbb{N}_+^d, x \in S, M_h^i(x) = U\right\}$ (for more on subshifts, see [13] for instance).

A *cellular automaton* is a quadruple $\langle A, d, r, g\rangle$, where A is the alphabet also called the *state set*, d is the dimension, $r \in \mathbb{N}$ is the *radius* and $g : \mathcal{M}^d_{\mathbf{2r+1}} \to A$ is the *local rule* of the automaton. The local rule g induces a *global rule* $G : A^{\mathbb{Z}^d} \to A^{\mathbb{Z}^d}$ defined as follows,

$$\forall x \in A^{\mathbb{Z}^d}, \forall i \in \mathbb{Z}^d, \quad G(x)_i = g\big(M^{i-\mathbf{r}}_{\mathbf{2r+1}}(x)\big) \enspace .$$

Note that CA are exactly the class of all shift-commuting functions which are (uniformly) continuous with respect to the Tychonoff metric (Hedlund's theorem from [10]). For the sake of simplicity, we will make no distinction between a CA and its global rule G.

The local rule g can be extended naturally to all finite matrices in the following way. With a little abuse of notation, for any $h \in [2r+1, +\infty)^d$ and any $U \in \mathcal{M}^d_h$, define $g(U)$ as the matrix obtained by the simultaneous application of g to all the $\mathcal{M}^d_{\mathbf{2r+1}}$ submatrices of U. Formally, $g(U) = M^{\mathbf{r}}_{h-\mathbf{2r}}(G(x))$, where x is any configuration such that $M^0_h(x) = U$.

2.2. SA Configurations

A *SA configuration* (or simply *configuration*) is a set of sand grains organized in piles and distributed all over the d-dimensional lattice $\mathbb{Z}^d$. A *pile* is represented either by an integer from $\mathbb{Z}$ (*number of grains*), or by the value $+\infty$ (*source of grains*), or by the value $-\infty$ (*sink of grains*), i.e., it is an element of $\widetilde{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$. One pile is positioned in each point of the lattice $\mathbb{Z}^d$. Formally, a configuration x is a function from $\mathbb{Z}^d$ to $\widetilde{\mathbb{Z}}$ which associates any vector $i = (i_1, \ldots, i_d) \in \mathbb{Z}^d$ with the number $x_i \in \widetilde{\mathbb{Z}}$ of grains in the pile of position i. Denote by $\mathcal{C} = \widetilde{\mathbb{Z}}^{\mathbb{Z}^d}$ the set of all configurations.

When the dimension d is known without ambiguity we note 0 the null vector of $\mathbb{Z}^d$ and $|i|$ the infinite norm of a vector $i \in \mathbb{Z}^d$. A *measuring device* β_r^m of precision $r \in \mathbb{N}$ and reference height $m \in \mathbb{Z}$ is a function from $\widetilde{\mathbb{Z}}$ to $\widetilde{[-r, r]}$ defined as follows

$$\forall n \in \widetilde{\mathbb{Z}}, \quad \beta_r^m(n) = \begin{cases} +\infty & \text{if } n > m + r \enspace , \\ -\infty & \text{if } n < m - r \enspace , \\ n - m & \text{otherwise.} \end{cases}$$

A measuring device is used to evaluate the relative height of two piles, with a bounded precision. This is the technical basis of the definition of cylinders, distances and ranges which are used all along this article.

In [4], the authors equipped $\mathcal{C}$ with a metric in such a way that two configurations are at small distance if they have the same number of grains in a finite neighborhood of the pile indexed by the null vector. The neighborhood is individuated by putting the measuring device at the top of the pile, if this latter contains a finite number of grains. Otherwise the measuring device is put at height 0. In order to formalize this distance, the authors introduced the notion of *cylinder*, that we rename *top cylinder*. For any configuration $x \in \mathcal{C}$, for any $r \in \mathbb{N}$, and for any $i \in \mathbb{Z}^d$, the top cylinder of x centered in i and of radius r is the d-dimensional matrix ${C'}^i_r(x) \in \mathcal{M}^d_{\mathbf{2r+1}}$ defined on the infinite alphabet $A = \widetilde{\mathbb{Z}}$ by

$$\forall k \in [1, 2r+1]^d,\ \left({C'}^i_r(x)\right)_k = \begin{cases} x_i & \text{if } k = r+1\ , \\ \beta_r^{x_i}(x_{i+k-r-1}) & \text{if } k \neq r+1 \text{ and } x_i \neq \pm\infty\ , \\ \beta_r^0(x_{i+k-r-1}) & \text{otherwise.} \end{cases}$$

In dimension 1 and for a configuration $x \in \mathcal{C}$, we have

$${C'}^i_r(x) = (\beta_r^{x_i}(x_{i-r}), \ldots, \beta_r^{x_i}(x_{i-1}), x_i, \beta_r^{x_i}(x_{i+1}), \ldots, \beta_r^{x_i}(x_{i+r}))$$

if $x_i \neq \pm\infty$, while

$${C'}^i_r(x) = \left(\beta_r^0(x_{i-r}), \ldots, \beta_r^0(x_{i-1}), x_i, \beta_r^0(x_{i+1}), \ldots, \beta_r^0(x_{i+r})\right)$$

if $x_i = \pm\infty$.

By means of top cylinders, the distance $\mathsf{d}' : \mathcal{C} \times \mathcal{C} \to \mathbb{R}_+$ has been introduced as follows:

$$\forall x, y \in \mathcal{C}, \quad \mathsf{d}'(x,y) = 2^{-k} \quad \text{where} \quad k = \min\left\{r \in \mathbb{N} : {C'}^0_r(x) \neq {C'}^0_r(y)\right\}\ .$$

Proposition 2.1 ([4, 6])**.** *With the topology induced by* d'*, the configuration space is locally compact, perfect and totally disconnected.* ■

2.3. Sand automata

For any integer $r \in \mathbb{N}$, for any configuration $x \in \mathcal{C}$ and any index $i \in \mathbb{Z}^d$ with $x_i \neq \pm\infty$, the *range* of center i and radius r is the d-dimensional matrix $R^i_r(x) \in \mathcal{M}^d_{\mathbf{2r+1}}$ on the finite alphabet $A = \widetilde{[-r, r]} \cup \perp$ such that

$$\forall k \in [1, 2r+1]^d, \quad \left(R^i_r(x)\right)_k = \begin{cases} \perp & \text{if } k = r+1\ , \\ \beta_r^{x_i}(x_{i+k-r-1}) & \text{otherwise.} \end{cases}$$

The range is used to define a sand automaton. It is a kind of top cylinder, where the observer is always located on the top of the pile x_i (called the *reference*). It represents what the automaton is able to see at position i. Sometimes the central $\perp$ symbol may be omitted for simplicity sake. The set of all possible ranges of radius r, in dimension d, is denoted by $\mathcal{R}^d_r$.

A *sand automaton* (SA) is a deterministic finite automaton working on configurations. Each pile is updated synchronously, according to a local rule which computes the variation of the pile by means of the range. Formally, a SA is a triple $\langle d, r, f\rangle$, where d is the dimension, r is the *radius* and $f : \mathcal{R}^d_r \to [-r, r]$ is the *local rule* of the automaton. By means of the local rule, one can define the *global rule* $F : \mathcal{C} \to \mathcal{C}$ as follows

$$\forall x \in \mathcal{C}, \forall i \in \mathbb{Z}^d, \quad F(x)_i = \begin{cases} x_i & \text{if } x_i = \pm\infty\ , \\ x_i + f(R^i_r(x)) & \text{otherwise.} \end{cases}$$

Remark that the radius r of the automaton has three different meanings: it represents at the same time the number of measuring devices in every dimension of the range (number of piles in the neighborhood), the precision of the measuring devices in the range, and the highest return value of the local rule (variation of a pile). It guarantees that there are only a finite number of ranges and return values, so that the local rule has finite description.

The following example illustrates a very simple sand automaton. For more examples, we refer to [6].

Example 2.2 (the automaton $\mathcal{N}$)**.** This automaton destroys a configuration by collapsing all piles towards the lowest one. It decreases a pile when there is a lower pile in the neighborhood (see Figure 1). Let $\mathcal{N} = \langle 1, 1, f_{\mathcal{N}} \rangle$ of global rule $F_{\mathcal{N}}$ where

$$\forall a, b \in \widetilde{[-1,1]}, \quad f_{\mathcal{N}}(a,b) = \begin{cases} -1 & \text{if } a < 0 \text{ or } b < 0 \ , \\ 0 & \text{otherwise.} \end{cases}$$

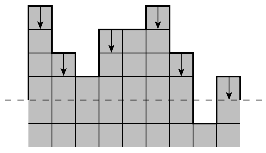


Figure 1: Illustration of the behavior of $\mathcal{N}$.

When no misunderstanding is possible, we identify a SA with its global rule F. For any $k \in \mathbb{Z}^d$, we extend the definition of the *shift map* to $\mathcal{C}$, $\sigma^k : \mathcal{C} \to \mathcal{C}$ is defined by $\forall x \in \mathcal{C}, \forall i \in \mathbb{Z}^d$, $\sigma^k(x)_i = x_{i+k}$. The *raising map* $\rho : \mathcal{C} \to \mathcal{C}$ is defined by $\forall x \in \mathcal{C}, \forall i \in \mathbb{Z}^d$, $\rho(x)_i = x_i + 1$. A function $F : \mathcal{C} \to \mathcal{C}$ is said to be *vertical-commuting* if $F \circ \rho = \rho \circ F$. A function $F : \mathcal{C} \to \mathcal{C}$ is *infinity-preserving* if for any configuration $x \in \mathcal{C}$ and any vector $i \in \mathbb{Z}^d$, $F(x)_i = +\infty$ if and only if $x_i = +\infty$ and $F(x)_i = -\infty$ if and only if $x_i = -\infty$.

Theorem 2.3 ([4, 6])**.** *The class of SA is exactly the class of shift and vertical-commuting, infinity-preserving functions $F : \mathcal{C} \to \mathcal{C}$ which are continuous w.r.t. the metric* d'. ■

3. Topology and dynamics

In this section we introduce a compact topology on the SA configuration space by means of a relation between SA and CA. With this topology, a Hedlund-like theorem still holds and each SA turns out to be homeomorphic to a CA acting on a specific subshift. We also characterize CA whose action on this subshift represents a SA. Finally, we study some topological properties of SA in this new setting.

3.1. A compact topology for SA configurations

From [6], we know that any SA of dimension d can be simulated by a suitable CA of dimension $d+1$ (and also any CA can be simulated by a SA). In particular, a d-dimensional SA configuration can be seen as a $(d+1)$-dimensional CA configuration on the alphabet $A = \{0,1\}$. More precisely, consider the function $\zeta : \mathcal{C} \to \{0,1\}^{\mathbb{Z}^{d+1}}$ defined as follows

$$\forall x \in \mathcal{C}, \quad \forall i \in \mathbb{Z}^d, \forall k \in \mathbb{Z}, \quad \zeta(x)_{(i,k)} = \begin{cases} 1 & \text{if } x_i \geq k \ , \\ 0 & \text{otherwise.} \end{cases}$$

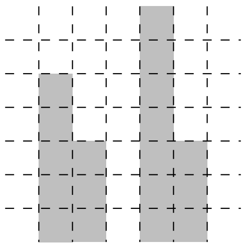
A SA configuration $x \in \mathcal{C}$ is coded by the CA configuration $\zeta(x) \in \{0,1\}^{\mathbb{Z}^{d+1}}$. Remark that ζ is an injective function.

Consider the $(d+1)$-dimensional matrix $K \in \mathcal{M}^{d+1}_{(}1, \ldots, 1, 2)$ such that $K_{1,\ldots,1,2} = 1$ and $K_{1,\ldots,1,1} = 0$. With a little abuse of notation, denote $S_K = S_{\{K\}}$ the subshift of configurations that do not contain the pattern K.

Proposition 3.1. *The set $\zeta(\mathcal{C})$ is the subshift S_K.*

Proof. Each d-dimensional SA configuration $x \in \mathcal{C}$ is coded by the $(d+1)$-dimensional CA configuration $\zeta(x)$ such that for any $i, h \in \mathbb{Z}^{d+1}, M_h^i(\zeta(x)) \neq K$, then $\zeta(\mathcal{C}) \subseteq S_K$. Conversely, we can define a preimage by ζ for any $y \in S_K$, by $\forall i \in \mathbb{Z}^d, x_i = \sup\{k : y_{(i,k)} = 1\}$. Hence $\zeta(\mathcal{C}) = S_K$. ∎

Figure 2 illustrates the mapping ζ and the matrix $K = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ for dimension $d = 1$. The set of SA configurations $\mathcal{C} = \widetilde{\mathbb{Z}}^{\mathbb{Z}}$ can be seen as the subshift $S_K = \zeta(\mathcal{C})$ of the CA configurations set $\{0,1\}^{\mathbb{Z}^2}$.


(a) Valid configuration.

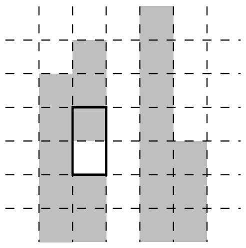

(b) Invalid configuration.

Figure 2: The configuration from Figure 2(a) is valid, while the configuration from Figure 2(b) contains the forbidden matrix K: there is a "hole".

Definition 3.2. The distance $\mathsf{d} : \mathcal{C} \times \mathcal{C} \to \mathbb{R}_+$ is defined as follows:

$$\forall x, y \in \mathcal{C}, \quad \mathsf{d}(x,y) = \mathsf{d}_T(\zeta(x), \zeta(y)) \ .$$

In other words, the (well defined) distance d between two configurations $x, y \in C$ is nothing but the Tychonoff distance between the configurations $\zeta(x), \zeta(y)$ in the subshift S_K. The corresponding metric topology is the $\{0,1\}^{\mathbb{Z}^{d+1}}$ product topology induced on S_K.

Remark 3.3. Note that this topology does not coincide with the topology obtained as countable product of the discrete topology on $\widetilde{\mathbb{Z}}$. Nevertheless, if you consider the topology $\mathcal{T}$ on $\widetilde{\mathbb{Z}}$ based on singletons $\{a\}$ where $a \in \mathbb{Z}$ and infinite intervals $[a, \infty]$ or $[-\infty, a]$, where $a \in \mathbb{Z}$, then d corresponds to its product topology. In other words, for any $i \in \mathbb{Z}^d$, the ith projection $\pi_i : \mathcal{C} \to \widetilde{\mathbb{Z}}$ defined by $\pi_i(x) = x_i$ is continuous for $\mathcal{T}$.

By definition of this topology, if one considers ζ as a map from $\mathcal{C}$ onto S_K, ζ turns out to be an isometric homeomorphism between the metric spaces $\mathcal{C}$ (endowed with d) and S_K (endowed with d_T). As an immediate consequence, the following results hold.

Proposition 3.4. *The set $\mathcal{C}$ is a compact and totally disconnected space where the open balls are clopen (i.e., closed and open) sets.* ∎

Proposition 3.5. *The space $\mathcal{C}$ is perfect.*

Proof. Choose an arbitrary configuration $x \in \mathcal{C}$. For any $n \in \mathbb{N}$, let $l \in \mathbb{Z}^d$ such that $|l| = n$. We build a configuration $y \in \mathcal{C}$, equal to x except at site l, defined as follows

$$\forall j \in \mathbb{Z}^d \setminus \{l\}\,,\ y_j = x_j \quad \text{and} \quad y_l = \begin{cases} 1 & \text{if } x_l = 0\ , \\ 0 & \text{otherwise.} \end{cases}$$

By Definition 3.2, $\mathsf{d}(y, x) = 2^{-n}$. ■

Consider now the following notion.

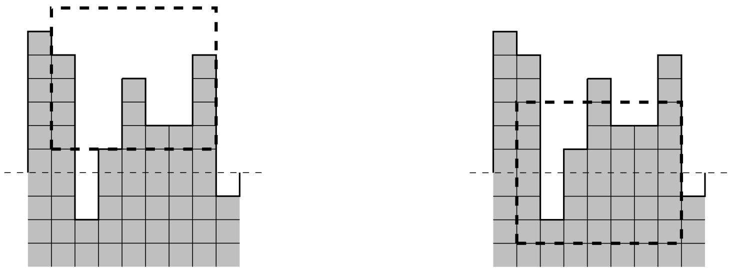
Definition 3.6 (ground cylinder)**.** For any configuration $x \in \mathcal{C}$, for any $r \in \mathbb{N}$, and for any $i \in \mathbb{Z}^d$, the *ground cylinder* of x centered on i and of radius r is the d-dimensional matrix $C_r^i(x) \in \mathcal{M}^d_{\mathbf{2r+1}}$ defined by

$$\forall k \in [1, 2r+1]^d\,, \quad \left(C_r^i(x)\right)_k = \beta_r^0(x_{i+k-r-1})\ .$$

For example in dimension 1,

$$C_r^i(x) = \left(\beta_r^0(x_{i-r}), \ldots, \beta_r^0(x_i), \ldots, \beta_r^0(x_{i+r})\right)\ .$$

Figure 3 illustrates top cylinders and ground cylinders in dimension 1. Remark that the contents of the two kinds of cylinders is totally different.



(a) Top cylinder centered on $x_i = 4$: $C'^i_r(x) = (+1, -\infty, -3, \mathbf{4}, -2, -2, +1)$.

(b) Ground cylinder, at height 0: $C_r^i(x) = (+\infty, -2, +1, +\infty, +2, +2, +\infty)$.

Figure 3: Illustration of the two notions of cylinders on the same configuration, with radius 3.

From Definition 3.2, we obtain the following expression of distance d by means of ground cylinders.

Remark 3.7. For any pair of configurations $x, y \in \mathcal{C}$, we have

$$\mathsf{d}(x, y) = 2^{-k} \quad \text{where} \quad k = \min\left\{r \in \mathbb{N} : C_r^0(x) \neq C_r^0(y)\right\}\ .$$

As a consequence, two configurations x, y are compared by putting boxes (the ground cylinders) at height 0 around the corresponding piles indexed by 0. The integer k is the size of the smallest cylinders in which a difference appears between x and y. This way of calculating the distance d is similar to the one used for the distance d', with the difference that the measuring devices and the cylinders are now located at height 0. This is slightly less intuitive than the distance d', since it does not correspond to the definition of the local rule. However, this fact is not an issue all the more since the configuration space is compact and the representation theorem still holds with the new topology (Theorem 3.11).

Finally, for a cylinder U, denote by $[U]_r = \{x \in \mathcal{C}, C_r^0(x) = U\}$ the *open ball* of radius 2^{-r} centered on U. We may write $[U]$ when the radius of the ball can be omitted.

3.2. SA as CA on a subshift

Let (X, m_1) and (Y, m_2) be two metric spaces. Two functions $H_1 : X \to X$, $H_2 : Y \to Y$ are (topologically) *conjugated* if there exists a homeomorphism $\eta : X \to Y$ such that $H_2 \circ \eta = \eta \circ H_1$.

We are going to show that any SA is conjugated to some restriction of a CA. Let F a d-dimensional SA of radius r and local rule f. Let us define the $(d+1)$-dimensional CA G on the alphabet $\{0,1\}$, with radius $2r$ and local rule g defined as follows (see [6] for more details). Let $M \in \mathcal{M}_{\mathbf{4r+1}}^{d+1}$ be a matrix on the finite alphabet $\{0,1\}$ which does not contain the pattern K. If there is a $j \in [r+1, 3r]$ such that $M_{(2r+1,\dots,2r+1,j)} = 1$ and $M_{(2r+1,\dots,2r+1,j+1)} = 0$, then let $R \in \mathcal{R}_r^d$ be the range taken from M of radius r centered on $(2r+1, \dots, 2r+1, j)$. See figure 4 for an illustration of this construction in dimension $d = 1$.

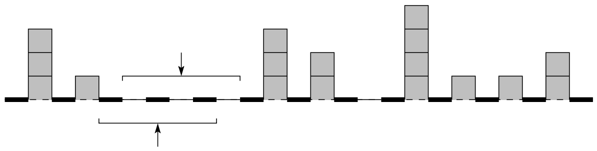


Figure 4: Construction of the local rule g of the CA from the local rule f of the SA, in dimension 1. A range R of radius r is associated to the matrix M of order $\mathbf{4r+1}$.

The new central value depends on the height j of the central column plus its variation. Therefore, define $g(M) = 1$ if $j + f(R) \geq 0$, $g(M) = 0$ if $j + f(R) < 0$, or $g(M) = M_{\mathbf{2r+1}}$ (central value unchanged) if there is no such j.

The following diagram commutes:

$$\begin{tikzcd} \mathcal{C} \arrow[r, "F"] \arrow[d, "\zeta"'] & \mathcal{C} \arrow[d, "\zeta"] \\ S_K \arrow[r, "G"'] & S_K \end{tikzcd} , \tag{3.1}$$

i.e., $G \circ \zeta = \zeta \circ F$. As an immediate consequence, we have the following result.

Proposition 3.8. *Any d-dimensional SA F is topologically conjugated to a suitable $(d+1)$-dimensional CA G acting on S_K.* ■

Being a dynamical submodel, SA share properties with CA, some of which are proved below. However, many results which are true for CA are no longer true for SA; for instance, injectivity and bijectivity are no more equivalent, as proved in [5]. Thus, SA deserve to be considered as a new model.

Corollary 3.9. *The global rule $F : \mathcal{C} \to \mathcal{C}$ of a SA is uniformly continuous w.r.t distance* d.

Proof. Let G be the global rule of the CA which simulates the given SA. Since the diagram (3.1) commutes and ζ is a homeomorphism, $F = \zeta^{-1} \circ G \circ \zeta$. The map G is continuous and, by Proposition 3.4, $\mathcal{C}$ is compact, which proves the corollary. ■

For every $a \in \mathbb{Z}$, let $\mathcal{C}_a = \pi_0^{-1}(\{a\})$ be the clopen (and compact) set of all configurations $x \in \mathcal{C}$ such that $x_0 = a$.

Lemma 3.10. *Let $F : \mathcal{C} \to \mathcal{C}$ be a continuous and infinity-preserving map. There exists an integer $l \in \mathbb{N}$ such that for any configuration $x \in \mathcal{C}_0$ we have $|F(x)_0| \leq l$.*

Proof. Since F is continuous and infinity-preserving, the set $F(\mathcal{C}_0)$ is compact and included in $\pi_0^{-1}(\mathbb{Z})$. From Remark 3.3, π_0 is continuous on the set $\pi_0^{-1}(\mathbb{Z})$ and in particular it is continuous on the compact $F(\mathcal{C}_0)$. Hence $\pi_0(F(\mathcal{C}_0))$ is a compact subset of $\widetilde{\mathbb{Z}}$ containing no infinity, and therefore it is included in some interval $[-l, l]$, where $l \in \mathbb{N}$. ∎

Theorem 3.11. *A mapping $F : \mathcal{C} \to \mathcal{C}$ is the global transition rule of a sand automaton if and only if all the following statements hold*

(i) F is (uniformly) continuous w.r.t the distance d;
(ii) F is shift-commuting;
(iii) F is vertical-commuting;
(iv) F is infinity-preserving.

Proof. Let F be the global rule of a SA. By definition of SA, F is shift-commuting, vertical-commuting and infinity-preserving. From Corollary 3.9, F is also uniformly continuous.

Conversely, let F be a continuous map which is shift-commuting, vertical-commuting, and infinity-preserving. By compactness of the space $\mathcal{C}$, F is also uniformly continuous. Let $l \in \mathbb{N}$ be the integer given by Lemma 3.10. Since F is uniformly continuous, there exists an integer $r \in \mathbb{N}$ such that

$$\forall x, y \in \mathcal{C} \quad C_r^0(x) = C_r^0(y) \Rightarrow C_l^0(F(x)) = C_l^0(F(y)) \ .$$

We now construct the local rule $f : \mathcal{R}_r^d \to [-r, r]$ of the automaton. For any input range $R \in \mathcal{R}_r^d$, set $f(R) = F(x)_0$, where x is an arbitrary configuration of $\mathcal{C}_0$ such that $\forall k \in [1, 2r+1]$, $k \neq r+1$, $\beta_r^0(x_{k-r-1}) = R_k$. Note that the value of $f(R)$ does not depend on the particular choice of the configuration $x \in \mathcal{C}_0$ such that $\forall k \neq r+1$, $\beta_r^0(x_{k-r-1}) = R_k$. Indeed, Lemma 3.10 and uniform continuity together ensure that for any other configuration $y \in \mathcal{C}_0$ such that $\forall k \neq r+1$, $\beta_r^0(y_{k-r-1}) = R_k$, we have $F(y)_0 = F(x)_0$, since $\beta_l^0(F(x)_0) = \beta_l^0(F(y)_0)$ and $|F(y)_0| \leq l$. Thus the rule f is well defined.

We now show that F is the global mapping of the sand automaton of radius r and local rule f. Thanks to (iv), it is sufficient to prove that for any $x \in \mathcal{C}$ and for any $i \in \mathbb{Z}^d$ with $|x_i| \neq \infty$, we have $F(x)_i = x_i + f\left(R_r^i(x)\right)$. By (ii) and (iii), for any $i \in \mathbb{Z}^d$ such that $|x_i| \neq \infty$, it holds that

$$\begin{aligned} F(x)_i &= \left[\rho^{x_i} \circ \sigma^{-i}\left(F(\sigma^i \circ \rho^{-x_i}(x))\right)\right]_i \\ &= x_i + \left[\sigma^{-i}\left(F(\sigma^i \circ \rho^{-x_i}(x))\right)\right]_i \\ &= x_i + \left[F(\sigma^i \circ \rho^{-x_i}(x))\right]_0 \ . \end{aligned}$$

Since $\sigma^i \circ \rho^{-x_i}(x) \in \mathcal{C}_0$, we have by definition of f

$$F(x)_i = x_i + f\left(R_r^0(\sigma^i \circ \rho^{-x_i}(x))\right) \ .$$

Moreover, by definition of the range, for all $k \in [1, 2r+1]^d$,

$$R_r^0(\sigma^i \circ \rho^{-x_i}(x))_k = \beta_r^{[\sigma^i \circ \rho^{-x_i}(x)]_0}(\sigma^i \circ \rho^{-x_i}(x)_k) = \beta_r^0(x_{i+k} - x_i) = \beta_r^{x_i}(x_{i+k}) \ ,$$

hence $R_r^0(\sigma^i \circ \rho^{-x_i}(x)) = R_r^i(x)$, which leads to $F(x)_i = x_i + f\left(R_r^i(x)\right)$. ∎

We now deal with the following question: given a $(d+1)$-dimensional CA, does it represent a d-dimensional SA, in the sense of the conjugacy expressed by diagram 3.1? In order to answer to this question we start to express the condition under which the action of a CA G can be restricted to a subshift $S_{\mathcal{F}}$, i.e., $G(S_{\mathcal{F}}) \subseteq S_{\mathcal{F}}$ (if this fact holds, the subshift $S_{\mathcal{F}}$ is said to be G-invariant).

Lemma 3.12. *Let G and $S_{\mathcal{F}}$ be a CA and a subshift of finite type, respectively. The condition $G(S_{\mathcal{F}}) \subseteq S_{\mathcal{F}}$ is satisfied iff for any $U \in \mathcal{L}(S_{\mathcal{F}})$ and any $H \in \mathcal{F}$ of the same order than $g(U)$, it holds that $g(U) \neq H$.*

Proof. Suppose that $G(S_{\mathcal{F}}) \subseteq S_{\mathcal{F}}$. Choose arbitrarily $H \in \mathcal{F}$ and $U \in \mathcal{L}(S_{\mathcal{F}})$, with $g(U)$ and H of the same order. Let $x \in S_{\mathcal{F}}$ containing the matrix U. Since $G(x) \in S_{\mathcal{F}}$, then $g(U) \in \mathcal{L}(S_{\mathcal{F}})$, and so $g(U) \neq H$. Conversely, if $x \in S_{\mathcal{F}}$ and $G(x) \notin S_{\mathcal{F}}$, then there exist $U \in \mathcal{L}(S_{\mathcal{F}})$ and $H \in \mathcal{F}$ with $g(U) = H$. ■

The following proposition gives a sufficient and necessary condition under which the action of a CA G on configurations of the G-invariant subshift $S_K = \mathcal{C}$ preserves any column whose cells have the same value.

Lemma 3.13. *Let G be a $(d+1)$-dimensional CA with state set $\{0,1\}$ and S_K be the subshift representing SA configurations. The following two statements are equivalent:*

(*i*) *for any $x \in S_K$ with $x_{(0,\ldots,0,i)} = 1$ (resp., $x_{(0,\ldots,0,i)} = 0$) for all $i \in \mathbb{Z}$, it holds that $G(x)_{(0,\ldots,0,i)} = 1$ (resp., $G(x)_{(0,\ldots,0,i)} = 0$) for all $i \in \mathbb{Z}$.*

(*ii*) *for any $U \in \mathcal{M}^d_{\mathbf{2r+1}} \cap \mathcal{L}(S_K)$ with $U_{(r+1,\ldots,r+1,k)} = 1$ (resp., $U_{(r+1,\ldots,r+1,k)} = 0$) and any $k \in [1, 2r+1]$, it holds that $g(U) = 1$ (resp., $g(U) = 0$).*

Proof. Suppose that (1) is true. Let $U \in \mathcal{M}^d_{\mathbf{2r+1}} \cap \mathcal{L}(S_K)$ be a matrix with $U_{(r+1,\ldots,r+1,k)} = 1$ and let $x \in S_K$ be a configuration such that $x_{(0,\ldots,0,i)} = 1$ for all $i \in \mathbb{Z}$ and $M^{-\mathbf{r}}_{\mathbf{2r+1}}(x) = U$. Since $G(x)_{(0,\ldots,0,i)} = 1$ for all $i \in \mathbb{Z}$, and $M^{0}_{\mathbf{2r+1}}(x) = U$, then $g(U) = 1$. Conversely, let $x \in S_K$ with $x_{(0,\ldots,0,i)} = 1$ for all $i \in \mathbb{Z}$. By shift-invariance, we obtain $G(x)_{(0,\ldots,0,i)} = 1$ for all $i \in \mathbb{Z}$. ■

Lemmas 3.12 and 3.13 immediately lead to the following conclusion.

Proposition 3.14. *It is decidable to check whether a given $(d+1)$-dimensional CA corresponds to a d-dimensional SA.* ■

3.3. Some dynamical behaviors

SA are very interesting models, whose complexity lies between that of d-dimensional and $d+1$-dimensional CA. Indeed, we have seen in the previous section that the latter can simulate SA, and it was shown in [6] that SA could simulate the former. A classification of one-dimensional cellular automata in terms of their dynamical behavior was given in [11]. Things appear to be very different as soon as we get into the second dimension, as noted in [15, 14]. This classification is based on the following notions.

Let (X, m) be a metric space and let $H : X \to X$ be a continuous application. An element $x \in X$ is an *equicontinuity* point for H if for any $\varepsilon > 0$, there exists $\delta > 0$ such that for all $y \in X$, $m(x,y) < \delta$ implies that $\forall n \in \mathbb{N}$, $m(H^n(x), H^n(y)) < \varepsilon$. The map H is *equicontinuous* if for any $\varepsilon > 0$, there exists $\delta > 0$ such that for all $x, y \in X$, $m(x,y) < \delta$ implies that $\forall n \in \mathbb{N}$, $m(H^n(x), H^n(y)) < \varepsilon$. An element $x \in X$ is *ultimately periodic* for H if there exist two integers $n \geq 0$ (the preperiod) and $p > 0$ (the period) such that $H^{n+p}(x) = H^n(x)$. H is *ultimately periodic* if there exist $n \geq 0$ and $p > 0$ such that $H^{n+p} = H^n$. H is *sensitive* (to the initial conditions) if there is a constant $\varepsilon > 0$ such that for all points $x \in X$ and all $\delta > 0$, there is a point $y \in X$ and an integer $n \in \mathbb{N}$ such that $m(x, y) < \delta$ but $m(F^n(x), F^n(y)) > \varepsilon$. H is *positively expansive* if there is a constant $\varepsilon > 0$ such that for all distinct points $x, y \in X$, there exists $n \in \mathbb{N}$ such that $m(H^n(x), H^n(y)) > \varepsilon$.

We consider these notions in the setting of sand automata with the metric topology induced by d. First we complete the definitions of equicontinuity and ultimate periodicity.

Proposition 3.15. *A SA F is equicontinuous if and only if all configurations of $\mathcal{C}$ are equicontinuity points.*

Proof. Suppose that all configurations are equicontinuity points. Let $\varepsilon > 0$. For all $x \in \mathcal{C}$, there is some δ_x such that for every configuration y such that $\mathsf{d}(x, y) < \delta_x$, we have $\forall n \in \mathbb{N}, \mathsf{d}(F^n(x), F^n(y)) < \frac{\varepsilon}{2}$. From the open covering $\mathcal{C} = \bigcup_{x\in\mathcal{C}} \left\{y | \mathsf{d}(x, y) < \frac{\delta_x}{2}\right\}$, we can extract a finite covering $\mathcal{C} = \bigcup_{x\in\mathcal{D}} \left\{y | \mathsf{d}(x, y) < \frac{\delta_x}{2}\right\}$, where $\mathcal{D} \subset \mathcal{C}$ is finite. Let $\delta = \min_{x\in\mathcal{D}} \frac{\delta_x}{2}$. Then for every $x, y \in \mathcal{C}$, such that $\mathsf{d}(x, y) < \delta$, there is some $z \in \mathcal{D}$ such that $\mathsf{d}(x, z) < \frac{\delta_z}{2}$. We also have $\mathsf{d}(y, z) < \delta + \frac{\delta_z}{2} \leq \delta_z$. Hence, for any $n \in \mathbb{N}$, $\mathsf{d}(F^n(x), F^n(y)) < \mathsf{d}(F^n(x), F^n(z)) + \mathsf{d}(F^n(y), F^n(z)) < \varepsilon$. Since this is true for any $\varepsilon > 0$, F is equicontinuous. The converse is trivial. ■

We introduce a helpful lemma, used to refine the notion of ultimate periodicity.

Lemma 3.16. *Any covering $\mathcal{C} = \bigcup_{k\in\mathbb{N}} \Sigma_k$ by closed shift-invariant subsets Σ_k contains $\mathcal{C} = \Sigma_k$ for some $k \in \mathbb{N}$.*

Proof. If $\mathcal{C} = \bigcup_{k\in\mathbb{N}} \Sigma_k$ where the Σ_k are closed, then by the Baire Theorem, some Σ_k has nonempty interior. Hence, it contains some ball $[U]$ where U is a cylinder. If it is shift-invariant, then it contains $\overline{\bigcup_{k\in\mathbb{Z}^d} \sigma^k([U])}$, which is the complete space. ■

Proposition 3.17. *A SA F is ultimately periodic if and only if all configurations of $\mathcal{C}$ are ultimately periodic points for F.*

Proof. Let F be a SA such that all configurations $x \in \mathcal{C}$ are ultimately periodic for F. For any $n \geq 0$ and $p > 0$, let $D_{n,p}$ be the closed shift-invariant subset $\{x : F^{n+p}(x) = F^n(x)\}$. Since $\mathcal{C} = \bigcup_{n,p\in\mathbb{N}} D_{n,p}$, by Lemma 3.16, $\mathcal{C} = D_{n,p}$ for some $n \geq 0$ and some $p > 0$. The converse is obvious. ■

Using the new compact topological framework, it is possible to prove that equicontinuity and ultimate periodicity are equivalent (proof in [7]).

Proposition 3.18 ([7])**.** *A SA is equicontinuous if and only if it is ultimately periodic.* ■

Despite these classical results, it appears that the classification from [11] into four classes (equicontinuous CA, non equicontinuous CA admitting an equicontinuity configuration, sensitive but not positively expansive CA, positively expansive CA) becomes irrelevant for one-dimensional SA. In particular, none of them satisfy the last topological concept of the classification (positive expansivity).

Proposition 3.19 ([7])**.** *There are no positively expansive SA.* ■

It also seems that the trichotomy between the other classes might be false. We conjecture that there exist non-sensitive SA without equicontinuity points, which would lead to another classification into four classes: equicontinuous, admitting an equicontinuity configuration (but not equicontinuous), non-sensitive without equicontinuity configurations, sensitive.

4. Conclusion

In this article we have continued the study of sand automata, by introducing a compact topology on the SA configurations set. In this new context of study, the characterization of SA functions of [4, 6] still holds. Moreover, a topological conjugacy of any SA with a suitable CA acting on a particular subshift might facilitate future studies about dynamical and topological properties of SA.

In particular, injectivity and surjectivity and their corresponding dimension-dependent decidability problems could help to understand if SA look more like CA of the same dimension or of the following one. Still in that idea is the open problem of the dichotomy between sensitive SA and those with equicontinuous configurations. A potential counter-example would give a more precise idea of the specificities of the dynamical behaviors represented by SA.

References

[1] E. Akin. *The general topology of dynamical systems.* Graduate Stud. Math. 1, AMS. Providence, 1993.

[2] P. Bak. *How nature works - The science of SOC.* Oxford University Press, 1997.

[3] P. Bak, C. Tang, and K. Wiesenfeld. Self-organized criticality. *Physical Review A*, 38(1):364–374, 1988.

[4] J. Cervelle and E. Formenti. On sand automata. In *Symposium on Theoretical Aspects of Computer Science (STACS 2003)*, volume 2607 of *Lecture Notes in Computer Science*, pages 642–653. Springer, 2003.

[5] J. Cervelle, E. Formenti, and B. Masson. Basic properties for sand automata. In *MFCS 2005*, volume 3618 of *Lecture Notes in Computer Science*, pages 192–211. Springer, 2005.

[6] J. Cervelle, E. Formenti, and B. Masson. From sandpiles to sand automata. *Theoretical Computer Science*, 381:1–28, 2007.

[7] A. Dennunzio, P. Guillon, and B. Masson. A compact topology for sand automata. arXiv, February 2008.

[8] E. Goles and M. A. Kiwi. Game on line graphs and sandpile automata. *Theoretical Computer Science*, 115:321–349, 1993.

[9] E. Goles, M. Morvan, and H. D. Phan. Sand piles and order structure of integer partitions. *Discrete Applied Mathematics*, 117:51–64, 2002.

[10] G. A. Hedlund. Endomorphisms and automorphisms of the shift dynamical system. *Mathematical Systems Theory*, 3:320–375, 1969.

[11] P. Kůrka. Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory & Dynamical Systems*, 17:417–433, 1997.

[12] P. Kůrka. *Topological and Symbolic Dynamics.* Volume 11 of Cours Spécialisés. Société Mathématique de France, 2004.

[13] D. Lind and B. Marcus. *An Introduction to Symbolic Dynamics and Coding.* Cambridge University Press, 1995.

[14] M. Sablik and G. Theyssier. Topological dynamics of 2D cellular automata. arXiv, September 2007.

[15] M. A. Shereshevsky. Expansiveness, entropy and polynomial growth for groups acting on subshifts by automorphisms. *Indag. Math.*, 4(2):203–210, jun 21 1993. N.S.

[16] R. Subramanian and I. Scherson. An analysis of diffusive load-balancing. In *ACM Symposium on Parallel Algorithms and Architecture (SPAA'94)*, pages 220–225. ACM Press, 1994.

Journées Automates Cellulaires 2008 (Uzès), pp. 228-237

STUDY OF THE NP-COMPLETENESS OF THE COMPACT TABLE PROBLEM

JEAN-CHRISTOPHE DUBACQ [1] AND JEAN-YVES MOYEN [1]

[1] LIPN/UMR 7030, CNRS/Université Paris XIII — 93430 Villetaneuse, France
URL: `http://www-lipn.univ-paris13.fr`

Abstract. The problem of compact tables is to maximise the overlap when building a word that is to include permutations of every given words (all the words being the same length). This problem is shown to be NP-complete in the general case, and some specific restrictions are studied.

1. Presentation of the problem

1.1. Informal description and examples

In several fields, it can be helpful to give a compact representation of a two dimensional table. We are interested in the case of random output tables, where the table is used to express the possible outcomes in n different initial conditions, where results can be described qualitatively with several results. These results can be represented several times for a given initial condition (to express discrete probabilities). We shall suppose that the number of possible outcomes (with multiple occurrences counted multiple times) is always the same (it could be 100 for a simplified percentage scheme, for example). This common number is called the amplitude of the table (ℓ).

For example, one can have the following table (on the left) where each row corresponds to a different set of initial conditions and each column corresponds to a different result. The numbers correspond to the possible outcomes, *i.e.* in case A, outcome α happens 30% of the time, outcome β happens 40% of the time and outcome γ happens with a probability of 30%. The same table can also be explicitly expanded (on the right).

	α	β	γ	δ
A	30%	40%	30%	0%
B	10%	30%	20%	40%
C	10%	0%	60%	30%
D	20%	20%	40%	20%

	1	2	3	4	5	6	7	8	9	10
A	α	α	α	β	β	β	β	γ	γ	γ
B	α	β	β	β	γ	γ	δ	δ	δ	δ
C	α	γ	γ	γ	γ	γ	γ	δ	δ	δ
D	α	α	β	β	γ	γ	γ	γ	δ	δ

Now, to find the outcome in, *e.g.* case B, one can computes a random number between 1 and 10 and directly look into the table for the corresponding result in the 'B' line.

2000 ACM Subject Classification: 68Q17, 68Q45, 91A90.
Key words and phrases: Probabilistic automaton, NP-completeness, Gaming theory.

If some of these results are common to several initial conditions, then it is possible to give a compact representation of the table. Instead of a $n \times \ell$ bi-dimensional table, we could express each initial condition as an offset in a one-dimensional table, that would cover all possible cases. It is always possible to do such a transformation as follows: give an offset of $i \times \ell$ to initial condition number i, and simply put all possible outcomes in a single table (in the order of the initial conditions). Initial condition 0 will use the 0 to $\ell - 1$ first possible outputs, initial condition 1 will use the ℓ to $2\ell - 1$ outputs, etc. It is quite obvious that the two formulations are equivalent.

The previous table can be "linearised" as follows :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
α	α	α	β	β	β	β	γ	γ	γ	α	β	β	β	γ	γ	δ	δ	δ	δ
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
α	γ	γ	γ	γ	γ	γ	δ	δ	δ	α	α	β	β	γ	γ	γ	γ	δ	δ

Then the result, *e.g* for case C can be directly read in the table by computing a random number between 21 and 30, or, rather, by computing a random number between 1 and 10 and adding an offset of 20.

Obviously, this linearised table is as big as the previous explicit table (40 results) and nothing have been gained that way.

The order of the initial conditions, however, is not important to the user of the table. The order of the outcomes is also not important. Therefore, one could move around initial conditions and outcomes so that some overlapping between results may appear.

Consider the following table :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
δ	δ	δ	δ	γ	γ	α	β	β	β	β	α	α	γ	γ	γ	γ	δ	δ	δ	γ	γ

Now, the result in each case can be read by computing a random number between 1 and 10 and adding an offset equal to:
A: +6, B: +0, C: +12, D: +9.

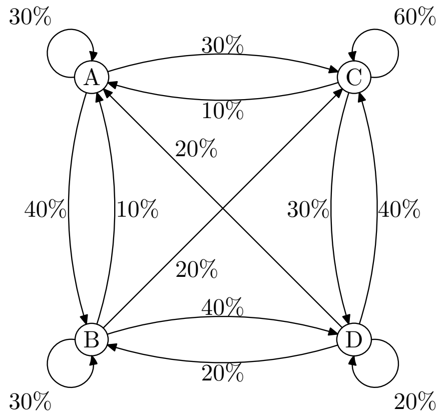
The resulting table is much smaller than the previous one (22 boxes instead of 40).

This kind of tables is, among other, used in simulation processes with simple models, such as the one you can find in *e.g.* wargames (historical simulation games). In wargames, a single action (such as a battle between armies) is usually resolved by rolling a die and looking for the result in a table. In this case, the initial conditions of the action (such as the relative strength of each army) determine the specific case and several outcomes (such as winning or loosing the battle with more or less casualties) can happen. To make things as unambiguous as possible between all players, it is better to have an explicit table rather than simply listing the probabilities of each outcome (which would be perfect in a computer-driven simulation but can lead to unnecessarily complicated computations as well as arguments over interpretation of the result in case of humans actually rolling dice). Obviously, compacting the tables allows to gain space and can thus be of great interest. Multi-dimensional tables are also very complex to remember (when several characteristics such as terrain, weather, strengths, etc. come into play in a single outcome) and linearised tables add to the playability in such games.

This is this kind of compact tables that this article studies (*e.g.* the assault table in [3]).

If the number of cases is less than or equal to the number of possible outcomes, this problem can be seen as the way of compacting transition tables for probabilistic automata. A probabilistic automaton is a normal (usually non-deterministic) automaton with weights on the various transitions; when a transition is made from some state, the probability a given transition will occur is given by the normalised weights of all possible transitions starting from the current state. If the normalised weights can be expressed as fractions with a common denominator, then the transition table of the probabilistic automaton can be expressed with partial overlaps as described above. In this case, the set of results and the set of initial conditions will be the same, namely the set of possible states of the automaton. For example, one can consider the previous table to be the transition table of a probabilistic automaton with 4 states. In state D, the automaton can goes to state A (result α) with probability 20%, in state B (result β) with probability 20% and so on.

The whole automaton can be depicted as follows:



The problem can also be seen as finding the shortest word containing a permutation of each of the words given as input. With our running example, there will be 4 words: $A = \alpha\alpha\alpha\beta\beta\beta\beta\gamma\gamma\gamma$, $B = \alpha\beta\beta\beta\gamma\gamma\delta\delta\delta\delta$, $C = \alpha\gamma\gamma\gamma\gamma\gamma\gamma\delta\delta\delta$ and $D = \alpha\alpha\beta\beta\gamma\gamma\gamma\gamma\delta\delta$ and there exists a word of length 22 containing a permutation of each of these words: $\delta\delta\delta\delta\gamma\gamma\alpha\beta\beta\beta\beta\alpha\alpha\gamma\gamma\gamma\gamma\delta\delta\delta\gamma\gamma$. Of course, the letters in each of the words (A, B, C, D) do not need to be ordered in each instance of the problem.

This leads to other applications to this problem: giving the shortest possible string that can contain permutations of a set of given strings may be of interest in the field of biology. A DNA molecule (or a strand of proteins) could be replicated several times, and separated in many sequences by a physical process. With simple weight analysis techniques, the composition (with no knowledge of the order of the components) of each string could be determined. From there, the shortest (and thus most likely) original string can be computed by looking for possible overlaps in the outcomes.

For example, consider a DNA molecule whose composition is unknown. Since the four bases have different weights[1], the weight of the molecule gives indication on its composition (especially if the molecule is small). However, this is not sufficient to precisely determine the molecule (that is, the order in which the bases appear in it).

By well known physical methods, it is possible to replicate the molecule several times. Then, each copy can be split into smaller parts. The parts can be sorted by weight using centrifugation. Hopefully, the parts will be small enough so that the exact composition of each part can be deducted from its weight (otherwise, one need to cut them again).

[1]Adenine weights 135.127g/mol, Thymine weights 126.113g/mol, Cytosine weights 111.300g/mol and Guanine weights 151.126g/mol.

Now, we have a set of words (the parts, whose composition is known) and we want to find a larger word (the whole molecule) such that it contains a permutation of each of the original words. The shortest solution is the more likely because a longer solution would probably have generated more different parts[2]. This is exactly the compact table problem.

1.2. Formal definition

The formal definition of the compact table problem is as follows:

Function Problem 1 (Compact table).

Instance: *An alphabet Σ, an integer ℓ, a set of words $S \subset \Sigma^\ell$*

Answer: *The minimal length k of a word $\tau \in \Sigma^k$ such that for any word $u \in S$, there exists a permutation σ and words v and w such that $\tau = v \cdot \sigma(u) \cdot w$.*

This problem is naturally associated to a decision problem, which we will show to be NP-complete in the general case.

Decision Problem 1 (Compact table).

Instance: *An alphabet Σ, an integer ℓ, a set of words $S \subset \Sigma^\ell$, an integer k*

Answer: Yes *if there exists a word $\tau \in \Sigma^k$ such that for any word $u \in S$, there exists a permutation σ and words v and w such that $\tau = v \cdot \sigma(u) \cdot w$,* No *in all other cases.*

2. General Case

To show that Compact Table is NP-complete, we shall first show that it is in the NP complexity class, and then that any instance of Hamiltonian Path can be transformed in an instance of Compact Table, such that the answer to the two problems is the same (Hamiltonian Path is a well-known NP-complete problem, see [10, 8]).

Decision Problem 2 (Hamiltonian path).

Instance: *A graph $G = (V, E)$ $(n = |V|)$*

Answer: Yes *if there exists a path $(v_1, e_1, v_2, e_2 \ldots, e_{n-2}, v_{n-1}, e_{n-1}, v_n)$ (such that $e_i = (v_i, v_{i+1})$) passing through all vertices of G,* No *if not.*

Compact Table is in NP:

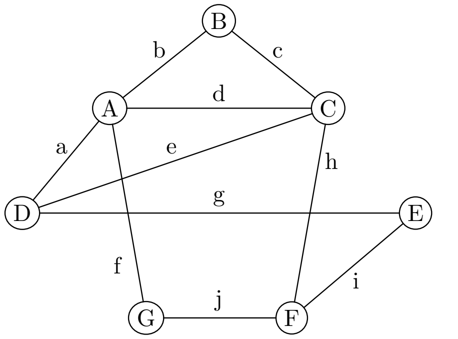
- The size of the inputs is no smaller than $|S| \times \ell$ because there are $|S|$ words of length ℓ each. The size of the result, τ, is smaller than $|S| \times \ell$ because there always exists a solution of length $|S| \times \ell$ (as shown in the examples above).
- Given τ (guessed non-deterministically), one first computes all the sub-words of τ of length ℓ (there are $|\tau| - \ell + 1$ such words) and compare each of them to each of the words in S (leading to $(|\tau| - \ell + 1) \times |S|$ comparisons between words).
- Each comparison between words can be done by first sorting the letters of the two words and then comparing the sorted words letter by letter. This require $O(\ell \log(\ell))$ comparisons for sorting and ℓ comparisons afterwards.
- Hence, the total number of comparisons (between letters) is $O(((|\tau| - \ell + 1) \times |S|) \times (\ell \log(\ell))) = O(|S|^2 \times \ell^2 \times \log(\ell))$.

[2]This means that the parts must not be too short, otherwise the same part may come from different places of the original molecule. However, this can be detected while weighting: if one part appear twice as many time as each other, then this is probably two identical parts.

Let us describe now how we shall transform an instance of HAMILTONIAN PATH in an instance of COMPACT TABLE. Let $G = (V, E)$ be a graph, ℓ be the maximum degree of G and $n = |V|$ be the number of vertices in G.

Construction: We define Σ to be the set $E \cup V$. Each vertex v is associated to a word τ_v of Σ^ℓ which is the set of edges adjacent to v (in no particular order) and completed (since G is not forced to be regular) by as many occurrences of v as deemed necessary. k is determined to be $n(\ell - 1) + 1$.

The following graph (on the left) is thus transformed into the words on the right:



- $\tau_A = abdf$
- $\tau_B = BBbc$
- $\tau_C = cdeh$
- $\tau_D = Daeg$
- $\tau_E = EEgi$
- $\tau_F = Fhij$
- $\tau_G = GGjf$

This graph admits a Hamiltonian path $ABCDEFG$ corresponding to the word (of length $n(\ell-1)+1 = 22$) $\tau = adf\mathbf{b}BB\mathbf{c}dh\mathbf{e}Da\mathbf{g}EE\mathbf{i}Fh\mathbf{j}GGf$; we build τ by choosing overlapping letters (shown in bold) corresponding to the edge linking consecutive vertices.

If τ exists and satisfies all conditions. We have to show that G admits a Hamiltonian path. Given the definition of the set S, the only way two words τ_v and $\tau_{v'}$ may overlap (even with permutations) is if v and v' share an edge (all other symbols are distinct). They can, at this point, overlap by one letter (edge (v, v')). The only way the final string τ can be of length $n(\ell-1)+1$ is if there are $n-1$ overlaps (all words have to be present, and this is a total of $n\ell$ letters). If this is the case, one can find a sequence of edges that join vertices and thus a sequence $(v_0, e_0, \dots, e_{n-2}, v_{n-1})$ of adjacent vertices and edges. This path passes through all vertices exactly once: if the string were redundant (one has not n but $m > n$ vertices), its length would be $m\ell - m + 1$. This is larger than k for all $m > n$. Thus, if τ exists, there exists a Hamiltonian path in G.

If G admits a Hamiltonian path. We have to show that there exists a word τ of length $n(\ell-1)+1$ that contains at least some permutation of the word associated to any vertex v of G. Consider one of the Hamiltonian paths $(v_1, e_1, v_2, e_2, \dots, v_{n-1}, e_{n-1}, v_n)$ and define τ as follows:

$$\begin{cases} \tau & = & u_1 \cdot e_1 \cdot u_2 \cdot e_2 \cdot u_3 \cdot \ldots \cdot u_n \\ u_i & = & \tau_{v_i} \text{ with } e_i \text{ and } e_{i-1} \text{ removed, } (1 < i < n) \\ u_1 & = & \tau_{v_1} \text{ with } e_1 \text{ removed} \\ u_n & = & \tau_{v_n} \text{ with } e_{n-1} \text{ removed} \end{cases}$$

It is quite straightforward that τ is of length $(n-2)(\ell-2) + 2(\ell-1) + (n-1)$, i.e. $n(\ell-2) + 2 + n - 1 = n(\ell-1) + 1$ and contains a permutation of τ_v for any vertex v (just write τ_v with the edge preceding v in the Hamiltonian path first and the edge following v in the Hamiltonian path last).

#P-completeness. We shall not introduce the counting problems versions of all these problems, but we mention the result here for the sake of completeness. The counting problem associated to HAMILTONIAN PATH is a #P-complete problem, as shown for example by theorem 18.2 of [13]. However, our transformation does not preserve the number of solutions (it is not *parcimonious*), and as such is not usable to determine whether COMPACT TABLE is a #P-complete problem. This is because of the allowed permutations: in our example above, the word $\tau' = daf\mathbf{b}BB\mathbf{c}dh\mathbf{e}Da\mathbf{g}EE\mathbf{i}Fh\mathbf{j}GGf$ (remark the difference in the beginning of the word) is another solution of the COMPACT TABLE instance, but is in correspondance to the exact same solution of HAMILTONIAN PATH. The number of solutions of COMPACT TABLE matching one precise HAMILTONIAN PATH solution is easy to compute (and they do not overlap), it is $(\ell-1)^2 \prod_{1\le i\le n} \dfrac{(\ell-2)!}{(\ell-d(i))!}$ (there are $\frac{(\ell-2)!}{(\ell-d(i))!}$ allowed permutations for the part of the word corresponding to each vertex, and $(\ell-1)$ times more for the initial/ending vertices). This number does however depend only on the chosen initial instance of HAMILTONIAN PATH, which means that COMPACT TABLE is #P-hard (if one can count the number of solutions of any instance of COMPACT TABLE, one can count the number of solutions of any instance of HAMILTONIAN PATH).

3. The fixed-amplitude case

3.1. Amplitude larger than 3

Let us consider the following family of decision problems (indexed by ℓ):

Decision Problem 3 (Compact table of order ℓ)**.**

Instance: *An alphabet Σ, a set of words $S \subset \Sigma^\ell$, an integer k*

Answer: YES *if there exists a word $\tau \in \Sigma^k$ such that for any word $u \in S$, there exists a permutation σ and words v and w such that $\tau = v \cdot \sigma(u) \cdot w$,* NO *in all other cases.*

It is obvious that the case $\ell = 1$ is polynomial. We can show that for any $\ell > 2$, COMPACT TABLE OF ORDER ℓ is still NP-complete, since the following family of problems is also NP-complete for all $d > 2$ (see [9, 12]):

Decision Problem 4 (Hamiltonian path in d-bounded degree graphs)**.**

Instance: *A graph $G = (V, E)$ ($n = |G|$) of maximal degree d*

Answer: YES *if there exists a path $(v_0, e_0, \ldots, e_{n-2}, v_{n-1})$ (such that $e_i = (v_i, v_{i+1})$) passing through all vertices of G,* NO *if not.*

The very same construction we used shows that for any $\ell > 2$, the COMPACT TABLE OF ORDER ℓ remains NP-complete.

3.2. Amplitude 2

We prove that in the case of amplitude 2, the problem becomes polynomial. In this case, each initial condition leads to an alternative between two results. We will reduce the problem to the following graph problem:

Function Problem 2 (Eulerian path)**.**

Instance: *An undirected $G = (V, E)$*

Answer: *A minimum set E' such that $G' = (V, E \cup E')$ is Eulerian.*

Let us consider an instance of COMPACT TABLE OF ORDER 2: let Σ be an alphabet and $S \subset \Sigma^2$ be a set of words[3] of length 2. Let $n = |S|$ be the number of words and $m = |\Sigma|$ be the number of letters of the alphabet.

We build a graph G with m vertices and n edges in the following way:

- There is a vertex α for each letter of the alphabet $\alpha \in \Sigma$.
- There is an edge between α and β if and only if the word $\alpha\beta$ is in S.

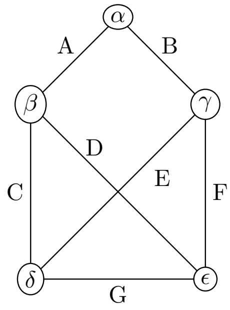
We shall now prove that the instance of COMPACT TABLE OF ORDER 2 admits a solution of length $n + k + 1$ if and only if G can be made Eulerian by adding k edges.

First, let G' be an Eulerian graph obtained by adding k edges to G. Consider an Eulerian path in G'. By enumerating the vertices traversed by the path, we obtain a word in Σ^* of length $n+k+1$ (there are $n+k$ edges, hence $n+k+1$ vertices). This word contains a permutation of each of the words in S because each of these words correspond to an edge in G, hence an edge in G', and the path has to go through all edges of G' by definition of Eulerian paths. Hence, if G can be made Eulerian by adding k edges to it, then COMPACT TABLE OF ORDER 2 admits a solution of length $n + k + 1$.

Conversely, if COMPACT TABLE OF ORDER 2 admits a solution τ of length $n + k + 1$. Let G'' be the smallest complete graph containing G. τ is the description of a path in G'' going through $n + k$ edges. Since τ contains a permutation of every word in S, all the edges of G belong to this path. Let G' be the graph obtained from G'' by keeping only the edges along this path. By construction, G' is Eulerian and contains all the edges in G plus k additional edges. Hence, if COMPACT TABLE OF ORDER 2 admits a solution of length $n + k + 1$, G can be made Eulerian by adding k edges to it.

For example, the table on the left (in each case, both results have the same probability) leads to the graph on the right:

	α	β	γ	δ	ϵ
A	1	1			
B	1		1		
C		1		1	
D		1			1
E			1	1	
F			1		1
G				1	1



Here, the graph is not Eulerian but can be made so by adding a single edge, H, between β and γ. This leads, among other, to the Eulerian path $EHABFGCD$ corresponding to the word $\delta\gamma\beta\alpha\gamma\epsilon\delta\beta\epsilon$ of length 9 with the offsets:

A: +2, B: +3, C: +6, D: +7, E: +0, F: +4, G: +5.

Let G be a graph. The minimum number of edges ones need to add to make it Eulerian[4] can be computed in polynomial time. Let us describe graphs as having c connected components and $2n$ vertices of odd degree. We separate the connected components between those with vertices of odd degree (A, $a = |A|$) and the others (B, $b = |B|$). A graph is

[3]We consider here that the words in S are all different. Identical words can be dealt with by considering multigraphs instead of graphs.

[4]The resulting graph may in fact be a multigraph, but this does not matter for our problem

Eulerian when (a, b, n) is $(1, 0, 1)$ or $(0, 1, 0)$. We shall suppose that we are not in one of those two cases.

If one edge is added to G, these are all the possible moves (we consider the variation of (a, b, n) as moves in a three-dimensional space):

- α**:** Adding one edge going from one component to itself: either $[0, 0, 1]$ between two even vertices, $[0, 0, 0]$ between an even vertex and an odd vertex, $[0, 0, -1]$ between two odd vertices. There is a special case for the last one: the move could also be $[-1, 1, -1]$.
- β**:** Adding one edge between two components of A: $[-1, 0, 1]$ between two even vertices, $[-1, 0, 0]$ between an even vertex and an odd vertex, $[-1, 0, -1]$ between two odd vertices.
- γ**:** Adding one edge between one component of A and one of B: $[0, -1, 1]$ if the vertex in the component in A was of even degree, $[0, -1, 0]$ otherwise. There is always an even number of odd-degree vertices in a component, so a never decreases this way.
- δ**:** Adding one edge between two components of B: $[1, -2, 1]$ (always).

The number of edges to be added (if not zero), is at least $b + n - 1$ (no move decreases $b + n$ by more than one, and the goal is of value 1). We exhibit a greedy (polynomial) algorithm that adds exactly $b + n - 1$ edges, and is therefore optimal.

- If $a = 0$, then $n = 0$ and $b > 1$. The transformation $\delta\gamma^{b-2}$ leads us to the final state and is of length $b + n - 1 = b - 1$.
- If $a > 0$, then transformation $\beta^{a-1}\gamma^{b}\alpha^{n-a}$ leads us to the final state and is of length $b + n - 1$.
- In each case, there is only one subcase that decreases $b + n$; there may be some choice for the exact edge to be added.

This is the smallest number of edges one must add to make the graph Eulerian as shown by Fleury's algorithm (see [4, 5, 6]). Remark that this is related to the Chinese postman problem [11] and the rural postman problem.

4. Limited number of results

4.1. The 2-results case

We would like to point out that the COMPACT TABLE OF ORDER ℓ problem with a limited number of possible results becomes trivial (solvable in constant time), because with a limited number of results and a fixed amplitude, the number of different words is finite.

Even the general COMPACT TABLE problem is polynomial in the case of only two possible results, namely success and failure (compare this to a toss of coin, with different probabilities of win according to some initial conditions). One can use a sequence of m_1 times the first result "0" followed by m_2 times the second result "1", where m_1 is the largest number of "0" for any initial condition and m_2 is the largest number of "1". Each initial condition can be associated to an offset that sets the number of possible "0" and "1". This word is the shortest possible since any satisfying word must have at least m_1 "0" and m_2 "1". For example, if the amplitude is 4 and the words are: $0000, 1101, 1001$ then the word 0000111 is the shortest one admitting permutations of 0000, 0111 and 0011 as continuous sub-words (with offsets 0, 3 and 2 respectively).

However, it should be remarked that giving a whole table for such a problem is not the best way to implement it. Since there are only two results, it is sufficient to give the amplitude and the offsets which correspond to the probabilities of "1", all other cases being "0". This means that the problem remains the same if, instead of words, one is simply given the number of "0" and "1" in each case (a more compact representation than then "unary" one used above).

This 2-results case is also of practical use. In genetic epidemiology, when introducing a new method to detect which genes can cause a disease, one has to test the method. This test is usually done on a simulated population. Each member of it is generated from a pre-existing pool of genotypes, respecting actual ratio observed in the population. Each member is then assigned a disease status (either affected or not). The probability of being affected depends on the genotype (corresponding to pre-existing observations made on the same disease, if the test validate the method, then it can later be carried over to not yet studied diseases) [2]. This corresponds exactly to the 2-results case: there are only two possible outcomes (affected or not) and the probability of each outcome depends on the input (the genotype).

4.2. With 3 or more results

For the case of a restricted alphabet (3 or more), the question is still open (about whether the problem remains NP-complete). However, the first remark still stands: if enough distinct words are given, all possible outputs finally appear. The number of such words (with k being the number of results and ℓ being the amplitude of the words) is $\binom{\ell+k-1}{\ell}$. A simple proof of this: a word is given by the number of occurrences of each result, including 0. This is in bijection to the words on alphabet $\{x, y\}$ of length $\ell + k - 1$, in which one chooses $k - 1$ delimiters "y" separating runs of "x" (a run can be of length 0). The distance between any two delimiters is the number of occurrences of the corresponding result (the first result at the beginning of the word, followed by the second, etc.).

However, the question of whether there exists a word that contains all possible combinations of length $\binom{\ell+k-1}{\ell} + \ell - 1$ (which would be the minimal length of such a word) is still open, and is too complex to fit in this paper. The authors conjecture that it is at least possible to do it for three results.

5. Conclusion and Future Works

We would like to remark that the similar albeit different problem where the initial words are given but no permutation is allowed to find them in the final word τ is also NP-complete in the general case.

Decision Problem 5 (Compact ordered table)**.**

Instance: *An alphabet Σ, an integer ℓ, a set of words $S \subset \Sigma^\ell$, an integer k*

Answer: YES *if there exists a word $\tau \in \Sigma^k$ such that for any word $u \in S$, there exist words v and w such that $\tau = v \cdot u \cdot w$,* NO *in all other cases.*

The associated function problem was shown to be NP-hard in [7], and several approximation have been shown since (see e.g. [1]).

We have restricted ourselves here to words of the same length (ℓ). Obviously, if the input words can be of any length the problem is more general, hence harder (the reduction from Hamiltonian Path still works) and the problem is also NP-complete (the proof of being in NP is the same as the one we present).

The question of whether or not the original problem is approximable is open. The heuristics for the compact ordered table may apply, but they are probably not very efficient. The originality of this work is that the permutations allowed might have been an ease to solve the problem, but being able to choose the order of overlapping sequences does not break the NP-hardness of the problem.

6. Thanks

We would like to thank Pierre Borgnat, for showing us the use of overlapping tables in wargames tables. We would like also to thank the anonymous referees for pointing out glaring mistakes, small typos, spelling errors found in the first draft and the interrogation about #P-completeness.

References

[1] Chris Armen and Clifford Stein. A 2 2/3 superstring approximation algorithm. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 88, 1998.

[2] Claire Bardel, Vincent Danjean, J.-P. Hugot, P. Darlu, and E. Génin. On the use of haplotype phylogeny to detect disease susceptibility loci. *BMC Genet.*, 6(1):24, May 2005.

[3] Pierre Borgnat, Bertrand Asseray, Jean-Yves Moyen, and Jean-Christophe Dubacq. Europa Universalis 8. `http://bamgames.free.fr/Europa/EU8/`, 2008.

[4] Reinhard Diestel. *Graph Theory*. Springer-Verlag, August 2005.

[5] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiæ scientiarum Petropolitanæ*, 8:128–140, 1741. Translations available in English.

[6] Henri Fleury. Deux problèmes de géométrie de situation. *Journal de Mathématiques élémentaires*, pages 257–261, 1883.

[7] John K. Gallant, David Maier, and James A. Storer. On finding minimal length superstrings. *Journal of Computer and System Sciences*, (20):50–58, 1980.

[8] Michael R. Garey and David S. Johnson. *Computers and Intractability*. Freeman, 1979.

[9] Michael R. Garey, David S. Johnson, and Robert E. Tarjan. The planar Hamiltonian circuit problem is NP-complete. *SIAM Journal of Computing*, 5(4):704–714, 1976.

[10] Richard Manning Karp. *Complexity of Computer Computations*, chapter Reducibility amoung Combinatorial Problems. Raymond E. Miller and James W. Thatcher, new york: plenum press edition, 1972.

[11] Mei-Ko Kwan. Graphic programming using odd or even points. *Chinese Math.*, (1):273–277, 1962.

[12] Maciej Liśkiewicz, Mitsunori Ogihara, and Seinosuke Toda. The complexity of counting self-avoiding walks in subgraphs of two-dimensional grids and hypercubes. *Theoretical Computer Science*, 304(1-3):129–156, 2003.

[13] Christos H. Papadimitriou. *Computational Complexity*, chapter 18, pages 439–443. Addison Wesley Longam, 1995.

Journées Automates Cellulaires 2008 (Uzès), pp. 238-249

THE SIGNAL POINT OF VIEW: FROM CELLULAR AUTOMATA TO SIGNAL MACHINES

Jérôme DURAND-LOSE [1]

[1] Laboratoire d'Informatique Fondamentale d'Orléans, Université d'Orléans, B.P. 6759, F-45067 ORLÉANS Cedex 2.
URL: `http://www.univ-orleans.fr/lifo/Members/Jerome.Durand-Lose`
E-mail address: `Jerome.Durand-Lose@univ-orleans.fr`

Abstract. After emphasizing on the use of discrete signals in the literature on cellular automata, we show how these signals have been considered on their own. We then present their continuous counterpart: abstract geometrical computation and signal machines. Finally we relate them to other models of computation, classical and analog.

Key-words. abstract geometrical computation, analog/continuous computation, black hole model, cellular automata, computability and signal machines.

1. Introduction

Cellular automata (CA) were introduced in the 1950's and have been used as a model for self-replication, computation, hardware, physics, economics... They are composed of identical automata, called *cells* displayed on a regular lattice and communicating only with neighbors. The dynamics, defined by the transition function of a cell, is parallel and synchronous. Different points of view are commonly used: considering a single cell or an entire configuration or the whole space-time diagram.

In the past decades, a new point of view has emerged: the cells are just a substrata on which information travels. The atomic pieces of information are signals: patterns that keep repeating regularly. The dynamics can then be understood as well as conceived in terms of signals.

In this paper, we show that this signal approach is quite usual both for describing CA generated from modeling and for designing special purpose CA. Then we recall some constructions on discrete signals and present signals in a continuous setting, *abstract geometrical computation* before stating some of their computing capabilities.

Signals are embedded inside a discrete structure: both space and time are discrete. On space-time diagrams they correspond more or less to discrete lines. On the one side, the granularity of space is often exploited to get a natural scale and to get a halting condition (for example to finish a Firing squad synchronization). On the other side it imposes a quite cumbersome setting: discrete geometry; for example halfway between two cells is either on a cell or between two cells.

To generate special purposed CA, usually an Euclidean setting is used for conception and then brute force and technical skills are used to discretize. The other way round, to explain dynamics, one often forgets about the discreteness and moves on to a continuous setting for an easier explanation. Both approaches rely on scaling invariance: if the granularity is thinner or thicker, the same phenomenon happens so that it does not depend on the number of cells.

These observations led us to consider the continuous case on its own. On the one hand, there is no problem with finding the middle; all positions are available. On the other hand, there is no granularity on which to rely for an absolute scale or to ensure a correct stop.

The discrete and continuous cases have similarities; for example, both can compute (in the classical sense) and thus have numerous related undecidability problems. Nevertheless, the continuous model is quite different and addresses different topics. For example, the signal approach have been very fruitful to solve the Firing squad synchronisation problem but the constructions lead to "monsters" on the continuous side (like accumulations on a Cantor set).

Signal machines can compute anything in the classical understanding as well as CA. In the continuous setting, Zeno effect (infinitely many steps in a finite duration) can appear and be used efficiently to decide semi-decidable problems, whereas in CA it is impossible. In another direction, since it works in a continuous setting it can handle real numbers with exact precision and be related to other analog models of computation like the Blum, Shub and Smale one.

The paper is articulated as follows. Section 2 briefly recalls what cellular automata, space-time diagrams and discrete signals are. Section 3 provides examples from the literature where signals are used a mean of explanation. Section 4 presents the approach and results on discrete signals, whereas Sect. 5 presents its continuous counterpart and some results on its computing capabilities. Conclusion and perspectives are gathered in Sect. 6.

2. Cellular automata

This section grounds the notations. Only CA with one dimension are addressed, almost everything naturally extends to higher dimensions.

Definition 2.1. A *cellular automaton* (CA) is defined by (Q, r, f) where Q is a finite set of *states*, the *radius*, r, is a positive integer, and the local function, f, is a function from Q^{2r+1} to Q. A *configuration* is an element of $Q^{\mathbb{Z}}$. The *global function*, $\mathcal{G} : Q^{\mathbb{Z}} \to Q^{\mathbb{Z}}$, is defined by:

$$\mathcal{G}(c)_i \;=\; f(c_{i-r}, c_{i-r+1} \dots c_i \dots c_{i-r-1}, c_{i-r}) \ .$$

Definition 2.2. A *space-time diagram*, $\mathcal{D}$, is the orbit of a configuration, c_0. It is an element of $Q^{\mathbb{Z}\times\mathbb{N}}$, $\mathcal{D}(.,0)$ is c_0 and $\mathcal{D}(i,t)$ is $\mathcal{G}^t(c)_i$.

Unless noted otherwise, space-time diagrams are represented with time increasing upward. Each configuration is set right above the previous one.

The following definition is empirical. It may vary according to articles and is more conceptual that formal.

Definition 2.3. A *background* is a state pattern that may legally tile a whole space-time diagram. A *signal* is a pattern that is periodically repeating over a background. Its *speed* is defined as the spacial shift divided by the number of iterations to repeat.

For example, if 0 is a quiescent state (*i.e.* $f(0,\ldots,0) = 0$) then a configuration filled with 0 is mapped onto itself and the resulting space-time diagram is filled with 0, so that 0 is a background. If the dynamics is such that a configuration with only 0's except for 1 on three consecutive cells, is regenerated every other iteration shifted by one cell on the right, then 111 is a signal and its speed is 1/2.

The speeds are bounded by the radius. This is a speed-of-light-like limitation. In space-time diagrams, the more a signal is vertical, the more it is slow. Signals have a *width*: the length of the pattern. The generated ones are generally 1-cell thin, but the ones observed are frequently not that thin.

3. Informal use of signals

Signals are information conveyors. The dynamics is driven by the information received, proceeded and sent.

3.1. Analysing the dynamics

3.1.1. *Particles and solitons.* Signals can be thought of as moving objects. This leads to the vocabulary of "particles". The term "soliton" is also used, but it implies that they can cross one another unaffected, like waves. Figure 1 shows some examples from the literature. This approach is important in physical modeling to ensure that studied objects could exist.

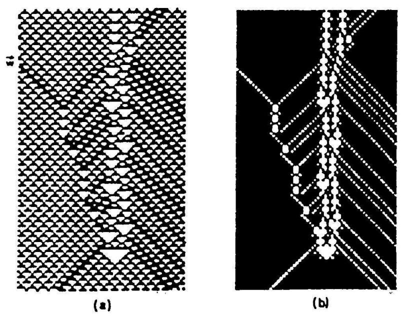


FIG. 7. Rule 54. (a) Annihilation of the radiating particle. (b) The same as (a) with the mapping defined in Fig. 6.
(a) [BNR91, Fig. 7]

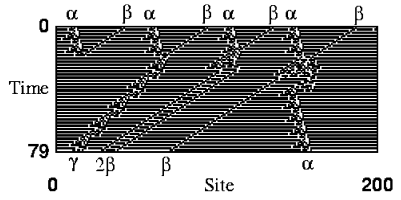


FIG. 7. The four different (out of 14 possible) interaction products for the $\alpha + \beta$ interaction.
(b) [HSC01, Fig. 7]

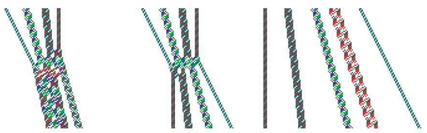


Figure 5. Two collisions of filtrons, and five free filtrons supported by the FPS model; ST diagram applies $q = 1$.
(c) [Siw01, Fig. 5]

(Time is increasing downward.)

Figure 1: Examples of particles and solitons.

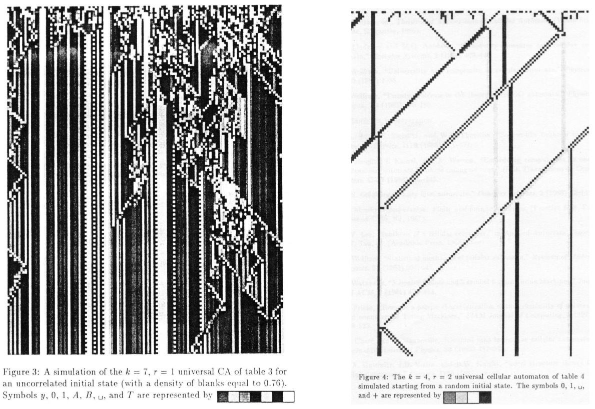


Figure 3: A simulation of the $k = 7$, $r = 1$ universal CA of table 3 for an uncorrelated initial state (with a density of blanks equal to 0.76). Symbols y, 0, 1, A, B, ␣, and T are represented by
Figure 4: The $k = 4$, $r = 2$ universal cellular automaton of table 4 simulated starting from a random initial state. The symbols 0, 1, ␣, and + are represented by

(Time is increasing downward.)

Figure 2: Signals to build an universal Turing machine [LN90, Fig. 3 and 4].

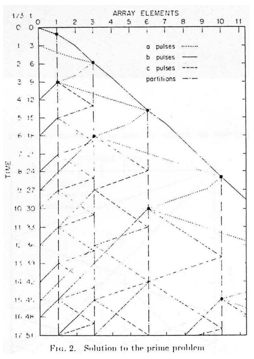
3.1.2. *Computing capability.* In computer science, before computing *better* (whatever it means), one is interested in being able to compute. In [LN90] (Fig. 2), signals are found so as to provide states and tape to simulate Turing machines.

In the quest for minimal Turing-universal and intrinsically universal cellular automata [Oll02, Coo04, Ric06], finding signals have often been the key to success.

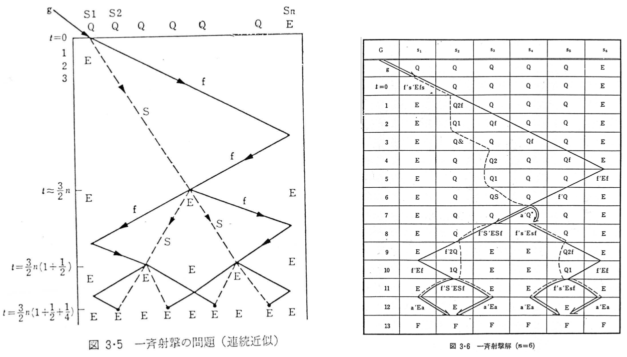
3.2. Generating particular CA

The other way round, signals have also been used to design special purpose CA.

3.2.1. *Prime number generation.* One application is to generate the prime numbers as the iteration numbers with no signal on cell 0 (*i.e.* on the leftmost vertical line) as done on Fig. 3(a) [Fis65]. Other sets of natural numbers can be enumerated this way [MT99].



(a) [Fis65, Fig. 2]


g
S1 S2
Sn
Q Q Q Q Q Q E
$t=0$
1
2
3
E
f
S
f
$t\approx\frac{3}{2}n$
E
f
f
E
E
S
S
$t=\frac{3}{2}n(1+\frac{1}{2})$
E
E
E
E
E
$t=\frac{3}{2}n(1+\frac{1}{2}+\frac{1}{4})$
E E E E E E E E E
図 3·5 一斉射撃の問題（連続近似）
図 3·6 一斉射撃解 (n=6)

G	s_1	s_2	s_3	s_4	s_5	s_6
g	Q	Q	Q	Q	Q	E
$t=0$	f′s′Efs	Q	Q	Q	Q	E
1	E	Q2f	Q	Q	Q	E
2	E	Q1	Qf	Q	Q	E
3	E	Q&	Q	Qf	Q	E
4	E	Q	Q2	Q	Qf	E
5	E	Q	Q1	Q	Q	f′Ef
6	E	Q	QS	Q	f′Q	E
7	E	Q	Q	a′Q*	Q	E
8	E	Q	f′S′ESf	f′s′Esf	Q	E
9	E	f′2Q	E	E	Q2f	E
10	f′Ef	1Q	E	E	Q1	f′Ef
11	E	f′S′ESf	E	E	f′s′Esf	E
12	a′Ea	E	a′Ea	a′Ea	E	a′Ea
13	F	F	F	F	F	F

(b) Goto's solution to FSS [Got66, Fig. 3+6]

(Time is increasing downward.)

Figure 3: Geometric algorithms.

3.2.2. *Firing squad synchronization (FSS).* This is a typical synchronisation problem from distributed computing. The aim is to have all processors do something special for the first time simultaneously. They have no way to broadcast, nor a common clock to refer to. This is thought of as a line of soldiers that must shoot synchronously but are not aware of their number and have very poor means of communication: each one can only communicate with the closest soldier on each side. Two soldiers are particularised: the first is a general that will start the process and the last who knows that he is the last. In CA modeling, the cells represent the soldiers.

Most solutions work on a divide and conquer scheme. For example, Goto's algorithm (Fig. 3(b)) cuts the line of soldiers in half and restarts the process synchronously on each side. When the granularity of space is reached they shoot. A careful management of odd/even pieces ensures that granularity is reached everywhere synchronously.

4. The world of discrete signals

4.1. Towards a definition

In the previous Section, we show that empirical notions of signals are used according to the context. It makes it natural to look at signals not as a tool but as a subject in itself. This is an important change of point of view. States and transitions are not considered to be the central place for the dynamics but rather some underlying layer, some byte code for a higher level language. Things are defined at the signals level, and then compiled into states and transitions. The compilation is more or less automatic depending what a signal is and what is expected from it.

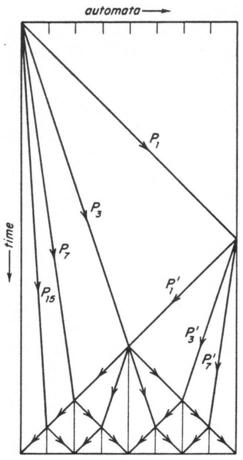
For example, in the FSS construction of [VMP70] (Fig. 4), infinitely many signals and speeds are considered. One would expect it hard if not impossible to bring this forth with finitely many states. This turns out to be possible, not only because speeds are bounded but also because, basically, the movement is managed by the interactions between signals. This family can be decomposed with a few bricks: a signal for moving, one for not moving together with signals ordering to move or to stop. Each signal forward the order to move only half of the time so that the second one is half slower, the next one is half of half…

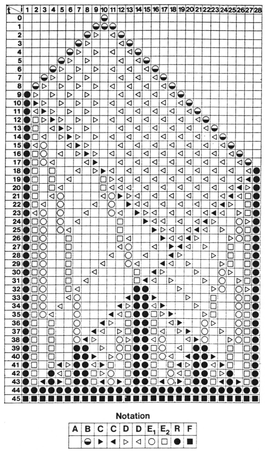
Considering this, one may think of some kind of jigsaw/tiling puzzle where thin pieces can be clipped on a board. Starting from the bottom, the board is filled upwards according to the way the pieces should be assembled together. This is the right level of abstraction. This can be implemented into CA and is abstract enough to design complex behaviours.

Let us propose a simple approach to compile signals of max speed 1 and width 1 (encoded on exactly one cell and with radius 1) to show its feasibility. Interaction only happens when signals are on the same cell. Signals are defined before their interactions.

A discrete signal is defined as a finite word over $\{\leftarrow, -, \rightarrow\}$ that corresponds to its periodic movements. For example, a word $\rightarrow$ would mean to move endlessly on the right, one cell at each iteration, whereas $-\rightarrow$ would mean right every other iteration. A signal does not need to be anything like a discrete line segment on a period (*e.g.* $\leftarrow\leftarrow\leftarrow\rightarrow\rightarrow\rightarrow$ is valid) although at a different scale it looks like a line.

Compilation is quite simple: in each cell there is a bit corresponding to each step of each signal. This means that every signal, at every step can be present in every cell! This generates a huge number of states. But signals can be handled very easily: if a signal is present with next move $-$, then it just goes to next step otherwise it is forgotten. If a signal





(Time is increasing downward.)

Figure 4: Geometrical Algorithm to solve the FSS [VMP70, Fig 1 and 3].

is present on the left (resp. right) with next move → (resp. ←), it goes on the next step on the current cell. Since signals are split into steps, this is well defined. Signals are endlessly moving.

Interactions/collisions can now be defined by rules like "if these signals are present at these steps, then they are replaced by those at those steps". Considering the vast amount of possibilities, one can imagine intended (and not extended) formulation and undefined cases could be handled by some superposition schemes and/or a default like: they cross unaffected or they disappear.

4.2. Achievements

This approach at the signal level together with an implementation (generally ad hoc and involving) has been quite fruitful: to give a improved solution to the FSS [Maz87], to design parabolas and circles [DMT99] and especially to develop a new kind of programing system with specific primitives.

For example, it is quite easy to have bits encoded by signals and have the dynamics carry out an addition or a multiplication [Maz96]. In these cases, the generated space-time diagrams look like the operation displayed as shown on Fig. 5.

There are ways to automatically have one computation twisted/bent so as to fit a portion of the diagram and to restart the computation in the room left. This produces recursion [DM02, Maz96, MT99] as displayed on Fig. 6.

5. Signal machines

In an euclidean space-time ($\mathbb{R} \times \mathbb{R}^+$), signals follow straight lines as illustrated on Fig. 7. They are dimension-less points (*i.e.* their width is zero). The dynamics of a single signal is not defined any more by a sequence of elementary displacements but by a constant speed. Thus its trace is a line segment in any space-time diagram.

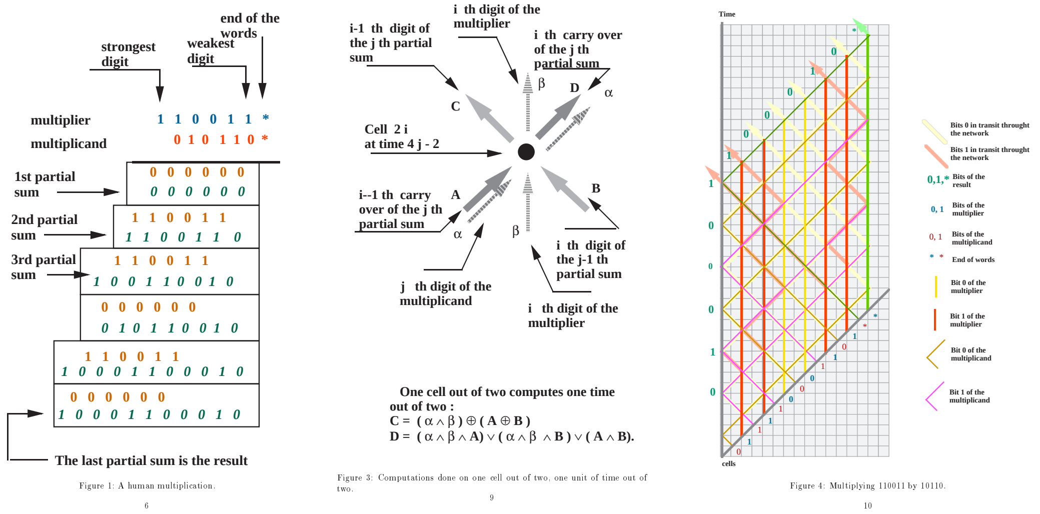

end of the words
strongest digit
weakest digit
multiplier 1 1 0 0 1 1 *
multiplicand 0 1 0 1 1 0 *
1st partial sum
2nd partial sum
3rd partial sum
The last partial sum is the result
Figure 1: A human multiplication.
i th digit of the multiplier
i-1 th digit of the j th partial sum
i th carry over of the j th partial sum
Cell 2 i at time 4 j - 2
i--1 th carry over of the j th partial sum
j th digit of the multiplicand
i th digit of the j-1 th partial sum
i th digit of the multiplier
One cell out of two computes one time out of two :
C = (α ∧ β) ⊕ (A ⊕ B)
D = (α ∧ β ∧ A) ∨ (α ∧ β ∧ B) ∨ (A ∧ B).
Figure 3: Computations done on one cell out of two, one unit of time out of two.
Time
cells
Bits 0 in transit throught the network
Bits 1 in transit throught the network
Bits of the result
Bits of the multiplier
Bits of the multiplicand
End of words
Bit 0 of the multiplier
Bit 1 of the multiplier
Bit 0 of the multiplicand
Bit 1 of the multiplicand
Figure 4: Multiplying 110011 by 10110.

Figure 5: Computation of a multiplication ([Maz96, Fig. 1, 3 andx 4]).

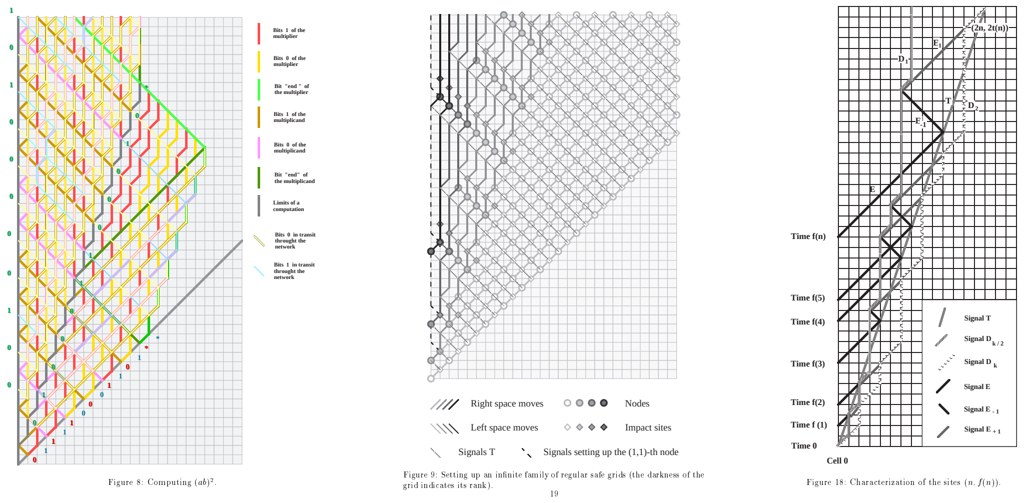

Bits 1 of the multiplier
Bits 0 of the multiplier
Bit "end" of the multiplier
Bits 1 of the multiplicand
Bits 0 of the multiplicand
Bit "end" of the multiplicand
Limits of a computation
Bits 0 in transit throught the network
Bits 1 in transit throught the network
Figure 8: Computing (ab)².
Right space moves
Left space moves
Signals T
Nodes
Impact sites
Signals setting up the (1,1)-th node
Figure 9: Setting up an infinite family of regular safe grids (the darkness of the grid indicates its rank).
(2n, 2t(n))
Time f(n)
Time f(5)
Time f(4)
Time f(3)
Time f(2)
Time f (1)
Time 0
Cell 0
Signal T
Signal D k/2
Signal D k
Signal E
Signal E -1
Signal E +1
Figure 18: Characterization of the sites (n, f(n)).

Figure 6: Geometric computing ([Maz96, Fig. 8 and 19], and [MT99, Fig. 18]).

The speed only depends on the nature of the signal since for discrete signals, it only depends on the pattern. Pragmatically, it simplifies everything; but nevertheless, the model is already very rich.

The whole dynamics is driven by signal collisions. When two or more signals meet, they are replaced by other signals according to some rules.

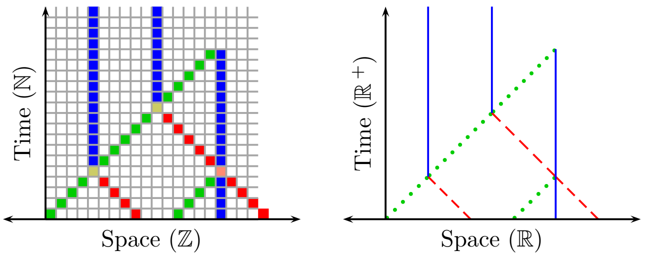


Figure 7: Space-time diagram of a cellular automaton and its signal machine counterpart.

5.1. Definition

Definition 5.1. A *signal machine* is defined by (M, S, R) where M is a finite set of meta-signals, S is a function $(M \to \mathbb{R})$ that assigns speeds, and R defines the collision rules (a function from 2^M to itself). A *signal* is an instance of a meta-signal.

There are finitely many signals which is not the case in some of above discrete examples. The reason is that otherwise the machine would not be finitely defined.

Let $\mu \in M$, if a μ-signal is at position x, then it will be at position $x + t.S(\mu)$ at time t if no other signal is met before.

A collision rule is denoted $\sigma^- \to \sigma^+$, if the meeting signals correspond to σ^-, they are removed and replaced by signals corresponding to σ^+. For example, in Fig. 7, whenever a dotted signal meets a dashed one they are replaced by a line one and a dashed one. Since R is a function, the dynamics is deterministic.

Definition 5.2. The *extended value set*, V, is the set of meta-signals plus two special values: $\oslash$ for the background (*i.e.* the absence of any signal) and ❊ for an accumulation. A *configuration* maps the underlying space to the extended set $(\mathbb{R} \to V)$ such that there are finitely many non $\oslash$ positions.

The finitely many non $\oslash$ signals condition amounts for the finiteness of a configuration ensuring that the collisions are clearly defined.

Definition 5.3. Let S_{min} and S_{max} be the minimal and maximal speeds. The *causal past*, or *(backward) light-cone*, arriving at position x and time t, $J^-(x,t)$, is defined by all the positions that might influence the information at (x,t) through signals, formally:

$$J^-(x,t) = \{\ (x',t') \mid x - S_{max}(t-t') \leq x' \leq x - S_{min}(t-t')\ \}\ .$$

The *space-time diagram* issued from an initial configuration c_0 and lasting for T, is a mapping c from $[0,T]$ to configurations (*i.e.* a mapping from $\mathbb{R} \times [0,T]$ to V) such that, $\forall (x,t) \in \mathbb{R} \times [0,T]$:

(1) $\forall t {\in} [0,T]$, $\{\, x \in \mathbb{R} \mid c_t(x) \neq \oslash \,\}$ is finite,

(2) if $c_t(x){=}\mu \in M$ then $\exists t_i, t_f {\in} [0,T]$ with $t_i{<}t{<}t_f$ or $0{=}t_i{=}t{<}t_f$ or $t_i{<}t{=}t_f{=}T$ s.t.:
- $\forall t' \in (t_i, t_f)$, $c_{t'}(x + S(\mu)(t'-t)) = \mu$,
- $t_i{=}0$ or $(\, c_{t_i}(x_i) = \rho^- {\to} \rho^+$ and $\mu \in \rho^+\,)$ where $x_i{=}x + S(\mu)(t_i{-}t)$,
- $t_f{=}T$ or $(c_{t_f}(x_f) = \rho^- {\to} \rho^+$ and $\mu \in \rho^-)$ or $c_{t_f}(x_f){=}$❊ where $x_f{=}x{+}S(\mu)(t_f{-}t)$;

(3) if $c_t(x){=}\rho^- {\to} \rho^+ \in R$ then $\exists \varepsilon$, $0{<}\varepsilon$, $\forall t' {\in} [t{-}\varepsilon, t{+}\varepsilon] \cap [0,T]$, $\forall x' {\in} [x-\varepsilon, x+\varepsilon]$,
- $(x',t') \neq (x,t) \Rightarrow c_{t'}(x') \in \rho^- {\cup} \rho^+ \cup \{\oslash\}$,
- $\forall \mu {\in} M$, $c_{t'}(x'){=}\mu \Leftrightarrow$ or $\begin{cases} \mu \in \rho^- \text{ and } t' < t \text{ and } x' = x + S(\mu)(t'-t))\,, \\ \mu \in \rho^+ \text{ and } t < t' \text{ and } x' = x + S(\mu)(t'-t))\,. \end{cases}$

(4) if $c_t(x) = ❄$ then $\forall\varepsilon > 0$, there is infinitely many signals in $J^-(x,t) \cap ([x-\varepsilon, x+\varepsilon] \times [t-\varepsilon, t])$.

Rules handle the collision of isolated signals. So that other kind of "continuation" would have to be defined when infinitely many signals are spatially accumulating to ensure that the configuration at $t + \epsilon$ is defined. Among the monsters of Fig. 8; there is Goto's FSS counterpart and a Cantor set generation. For CA, granularity ensures the correct achievement of the FSS, but there is no such thing here.

There is a tentative definition for the first kind of accumulation (leftmost case of Fig. 8) in [DL03, Chap. 9], simplified versions are used later on (nothing or a single signal is issued).

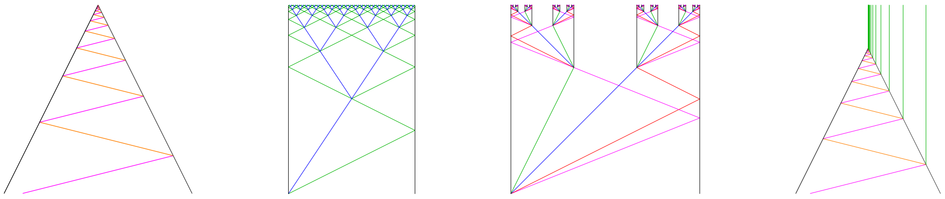


Figure 8: A simple accumulation and three unwanted phenomena.

5.2. Turing computing capabilities

Although Abstract geometrical computation relies on exact real values, with a simple restriction, it falls into the setting of classical computability. A signal machine is *rational* if it has only rational speeds and positions in any initial configuration. It is easy to see that, as long as there is no accumulation, all collisions happen at rational locations. Since rational numbers can be encoded and manipulated with exact precision on any computer (and the machine is finitely defined and there are finitely many signals in any configuration), implementation is possible (and has been done in Java to generate the illustrations). So that relating to Turing machine or any equivalent model makes sense.

In [DL05], it is proved that (rational) signal machine can simulate any counter automaton and thus have Turing power. This is still the case when only signal machines that are conservative and reversible are considered [DL06c]. Conservative means that each meta-signal has an positive energy and that each collision preserves this energy, so that the number of signals is bounded from the beginning. Reversible means that the collision rule is a bijection and that the signal machine can be run backward deterministically.

With computing capability comes undecidability, in the rational context many problems can be expressed in the classical setting. Some prediction problems (*e.g.* the apparition of a signal, the extension of a configuration on the side) are straightforwardly undecidable [DL05] (this is not surprising since there are many such results as well as a Rice theorem for CA [Kar94]). Collision forecasting is not even semi-decidable, it is Σ^2_0-complete in the arithmetical hierarchy [DL06b].

Let us illustrate the computing capability as well as available geometrical operations with the proof of Σ^2_0-hardness. This is done by reducing the (Π^2_0-complete) Total problem: whether a computable function (as defined by a 2-counter automaton for example) is defined for all values. On the left of Fig. 9 there is a simulation of a two-counter automaton (the vertical lines represent the counters). It is possible to add signals and rules allowing a computation to go on while being bent one way then bent back producing a scaling by one-half. This superstructure can restart itself and iterates infinitely (scaling is done three times in the middle space-time diagram of Fig. 9). This computation is more and more compressed and accelerated by a fractal structure it is entangled in. Since each iteration corresponds to the same uncompressed duration, the embedded computation has an infinite time ahead of it. If the computation stops (as in the picture), the structure and computation is erased so that there is no accumulation, otherwise the accumulation takes place (Π^1_0-hardness is already reached by reducing Halt).

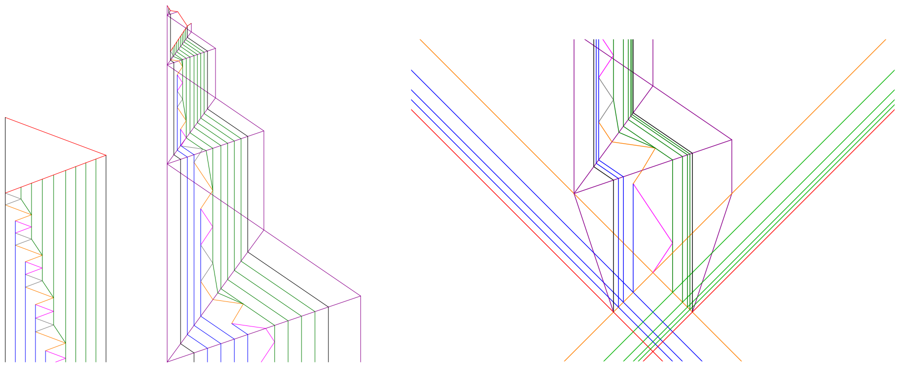


Figure 9: 2-counter automaton simulation, straight and contracted, and starting it.

On the right of Fig. 9, it is shown the way the value is provided and the computation started. On Fig. 10, a lattice is formed in order to try all the values of the counters.

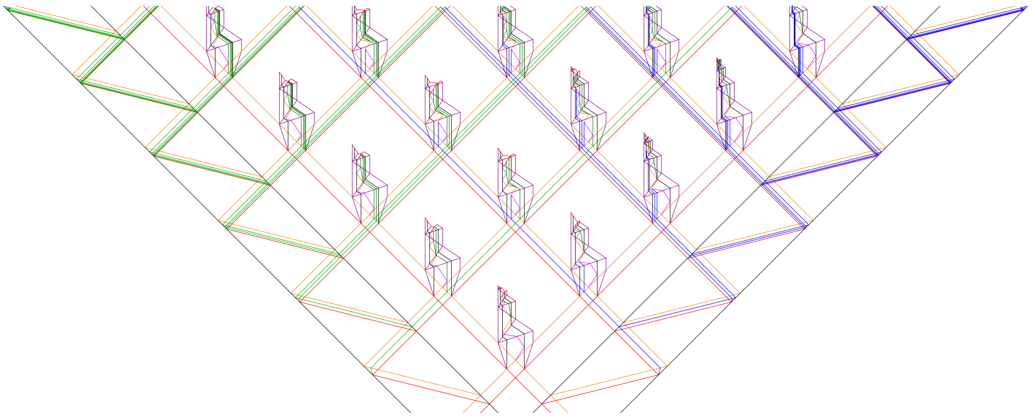


Figure 10: Trying all values.

With a proper use of simple accumulations (they just disappear leaving no trace), the so-called Black hole model of computation can be embedded inside rational signal machines and semi-decidable problems become decidable [DL06a]. The construction is as follows. The contracting computation is bounded by two signals. If the computation stops, a signal amounting for the answer is emitted and collected by the signals outside. When these two signals meet, it they had not received any signal they can assert that the computation never stopped.

5.3. Relations with the Blum-Shub-Smale model [BSS89].

Since the positions and speeds are any real number, computability issues refer to analog computation/computing on the continuous. Yet there is no analog Turing thesis and various incomparable definitions exist. Abstract geometrical computation has already been related to the BSS model: it is like a register machine where registers hold real numbers with exact addition, multiplication and sign test. Indirect addressing and infinitely many registers are available. Signal machines without any accumulation are equivalent to linear BSS, *i.e.* with the restriction that multiplication can only be by a constant [DL07]. The encoding of a linear BSS is done with constants in speeds and any real numbers held by a register as the distance between two parallel signals.

To achieve the full BSS, *i.e.* with internal multiplication, accumulations can be used [DL08]. Here accumulations results in a single signal where the accumulation takes place. Accumulations are used to compute infinite sums. The multiplication of two real numbers a and b is done by summing the products of a by the positive and negative powers of 2 according to the infinite binary expansion of b.

The resulting model is strictly more powerful than BSS since it can also performs square rooting. The formula/program for computing it uses only rational numbers so that the speeds of the machine are rational numbers. To compute the square root of 2, all signals are at rational positions. But the accumulation happens at the irrational position $\sqrt{2}$.

6. Conclusion

Abstract geometrical computation naturally arose from CA and is rich and promising.

The discrete constructions perfectly fit into the discrete word of CA as the continuous constructions perfectly fit the continuous word of SM. It would be interesting to investigate on how and in which cases the continuous side is a limit of the discrete one and the other way round, up to what amount can CA represent approximations of AGC. For example, on what conditions could there be an automatic discretisation of signal machine into CA in order to preserve some kind of properties?

Abstract geometrical computation still have to be studied on its own as well as related to computable analysis for its continuous aspects on one side and to transfinite computation since accumulation of order 2 and higher can be generated on the other side.

References

[BNR91] N. Boccara, J. Nasser, and M. Roger. Particle-like structures and interactions in spatio-temporal patterns generated by one-dimensional deterministic cellular automaton rules. *Phys. Rev. A*, 44(2):866–875, 1991.

[BSS89] L. Blum, M. Shub, and S. Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc.*, 21(1):1–46, 1989.

[Coo04] M. Cook. Universality in elementary cellular automata. *Complex Systems*, 15:1–40, 2004.

[DL03] J. Durand-Lose. *Calculer géométriquement sur le plan – machines à signaux.* Habilitation à diriger des recherches, École Doctorale STIC, Université de Nice-Sophia Antipolis, 2003.

[DL05] J. Durand-Lose. Abstract geometrical computation: Turing-computing ability and undecidability. In B. S. Cooper, B. Löwe, and L. Torenvliet, editors, *New Computational Paradigms, 1st Conference on Computability in Europe (CiE '05)*, number 3526 in LNCS, pages 106–116. Springer, 2005.

[DL06a] J. Durand-Lose. Abstract geometrical computation 1: embedding black hole computations with rational numbers. *Fundamenta Informaticae*, 74(4):491–510, 2006.

[DL06b] J. Durand-Lose. Forcasting black holes in abstract geometrical computation is highly unpredictable. In J.-Y. Cai, S. B. Cooper, and A. Li, editors, *Theory and Appliacations of Models of Computations (TAMC '06)*, number 3959 in LNCS, pages 644–653. Springer, 2006.

[DL06c] J. Durand-Lose. Reversible conservative rational abstract geometrical computation is turing-universal. In A. Beckmann and J. V. Tucker, editors, *Logical Approaches to Computational Barriers, 2nd Conference on Computability in Europe (CiE '06)*, number 3988 in LNCS, pages 163–172. Springer, 2006.

[DL07] J. Durand-Lose. Abstract geometrical computation and the linear Blum, Shub and Small model. In S. Cooper, B. Löwe, and A. Sorbi, editors, *Computation and Logic in the Real World 3rd Conference on Computability in Europe (CiE '07)*, number 4497 in LNCS, pages 238–247. Springer, 2007.

[DL08] J. Durand-Lose. Abstract geometrical computation: beyond the Blum, Shub and Small model with accumulation. Research Report RR-2008-01, LIFO, U. D'Orléans, France, 2008. http://www.univ-orleans.fr/lifo/prodsci/rapports.

[DM02] M. Delorme and J. Mazoyer. Signals on cellular automata. In A. Adamatzky, editor, *Collision-based computing*, pages 234–275. Springer, 2002.

[DMT99] M. Delorme, J. Mazoyer, and L. Tougne. Discrete parabolas and circles on 2D cellular automata. *Theoret. Comp. Sci.*, 218(2):347–417, 1999.

[Fis65] P. C. Fischer. Generation of primes by a one-dimensional real-time iterative array. *J. ACM*, 12(3):388–394, 1965.

[Got66] E. Goto. Ōtomaton ni kansuru pazuru [Puzzles on automata]. In T. Kitagawa, editor, *Jōhōkagaku eno michi [The Road to information science]*, pages 67–92. Kyoristu Shuppan Publishing Co., Tokyo, 1966.

[HSC01] W. Hordijk, C. R. Shalizi, and J. P. Crutchfield. An upper bound on the products of particle interactions in cellular automata. *Phys. D*, 154:240–258, 2001.

[Kar94] J. Kari. Rice's theorem for the limit sets of cellular automata. *Theoret. Comp. Sci.*, 127:229–254, 1994.

[LN90] K. Lindgren and M. G. Nordahl. Universal computation in simple one-dimensional cellular automata. *Complex Systems*, 4:299–318, 1990.

[Maz87] J. Mazoyer. A 6-states minimal-time solution to the Firing squad synchronisation problem. *Theoret. Comp. Sci.*, 50(2):183–237, 1987.

[Maz96] J. Mazoyer. Computations on one dimensional cellular automata. *Ann. Math. Artif. Intell.*, 16:285–309, 1996.

[MT99] J. Mazoyer and V. Terrier. Signals in one-dimensional cellular automata. *Theoret. Comp. Sci.*, 217(1):53–80, 1999.

[Oll02] N. Ollinger. The quest for small universal cellular automata. In *ICALP '02*, number 2380 in LNCS, pages 318–329. Springer, 2002.

[Ric06] G. Richard. A particular universal cellular automaton. oai:hal.ccsd.cnrs.fr:ccsd-00095821_v1, 2006.

[Siw01] P. Siwak. Soliton-like dynamics of filtrons of cycle automata. *Inverse Problems*, 17:897–918, 2001.

[VMP70] V. I. Varshavsky, V. B. Marakhovsky, and V. A. Peschansky. Synchronization of interacting automata. *Math. System Theory*, 4(3):212–230, 1970.

Journées Automates Cellulaires 2008 (Uzès), pp. 250-261

AN EXHAUSTIVE EXPERIMENTAL STUDY OF SYNCHRONIZATION BY FORCING ON ELEMENTARY CELLULAR AUTOMATA

JEAN-BAPTISTE ROUQUIER [1]

[1] Université de Lyon, ENS Lyon and IXXI,
LIP, 46 allée d'Italie,
69364 LYON, France

Abstract. We study a way of coupling two configurations of the same cellular automaton rule for all elementary cellular automata (ECA). We experimentally show that there are only two possible behaviors: either synchronization for all coupling strength, or a phase transition. This transition is shown to belong to the directed percolation universality class, even for a non chaotic rule and for rules with particles.

Introduction

Chaotic systems, with an apparently random behavior, have received a great deal of attention in the last decades. It might be expected that, when adding noise to the system, the behavior becomes even more chaotic. However, in various systems, a transition from chaotic to non-chaotic behavior has been observed when varying a parameter: for some values of the parameter, all trajectories become identical after a while and the system is no more chaotic, since trajectories are independent of initial conditions. Synchronization is made possible by the fact that all instances of the system are subject to the same realization of the noise. This idea if synchronizing systems by identical random perturbation can be traced back to [9] or [11].

The synchronization of simple dynamical systems taking cellular automata (CA) as a model is the subject of a survey in [15], together with other extended dynamical systems. The same authors in [10] describe a stochastic synchronization technique for CA: one considers two configurations initialized independently and randomly. Both follow the same CA rule. To try to synchronize them, one compares both configurations at each time step, cell by cell. If both cells differ, they are made equal with probability p. The set of cells that are made equal, or synchronized, is thus random and changing at each time step. The parameter p controls the strength of the synchronization.

2000 ACM Subject Classification: F: Theory of Computation / F.1: Computation by Abstract Devices / F.1.1: Models of Computation / F.1.1.5: Unbounded-Action Devices (e.g., Cellular Automata, circuits, networks of machines).

Key words and phrases: coalescence, coupling, robustness, discrete dynamical system, stochastic process, directed percolation, universality class, phase transition.

The goal of this paper is to explore this emergent phenomenon on all ECA (elementary CA, that is, CA with one dimension, two states and two nearest neighbors). Preceeding papers studied only one or two rules, chosen for the ease of simulation or for known chaotic properties. We show that one does not need a chaotic CA to observe interesting behavior (in this case, a phase transition) in this setting.

We study by simulation the behavior of all ECA with regard to this synchronization scheme and show that over the 88 different ECA, two behaviors are possible: synchronization even with the slightest synchronization strength, or synchronization only above a certain synchronization strength. For the second class, we study the transition between non synchronization and synchronization when p varies. There is always a phase transition belonging to the universality class of directed percolation. We thus get a new model of directed percolation, with a few variants. Like a few other directed percolation models [6, 10, 12], the limit of the sub-critical regime is neither a single absorbing state, nor a set of fixed points, but a non trivially evolving phase.

Perturbation. The model presented here is a kind of perturbation to the original CA. The general idea behind perturbation is to study the robustness of the system. Real systems are not as regular and defect-free as models, so a good predictive model has to be robust to small perturbation. We here study one kind of perturbation and apply it to all ECA.

Perturbing a system is a first step towards controlling it, indeed, some systems are controlled with small, carefully chosen perturbations (e.g. satellite trajectories). The perturbation studied here is a kind of "self-perturbation", in the sense that it is a perturbation induced by a CA following the same rule. Such a perturbation, with an external force that is related to the system of interest, might be more relevant than random noise.

Directed percolation. Directed percolation is found in other variations of the CA model. The authors of [2] have studied a continuous model that collapses to deterministic CA dynamics. They show that the observed synchronization transition, on changing the strength of the stochastic coupling between replicas, belongs to the directed percolation universality class. In [12], we presented another way of coupling two configurations of the same CA rule, called coalescence. For some rules, we observed that there is a phase transition between coalescence (the coupling makes both configurations equal) and non coalescence. As predicted by a conjecture from Grassberger [5], the transition belonged to the universality class of directed percolation. The conditions for this conjecture also apply to almost all rules of the present study, and this paper shows that the new model also belongs to this universality class.

The paper is organized as follows. Section 1 gives definitions, notations, and a few remarks. We describe the exhaustive simulation study in Section 2.1, then introduce directed percolation in Section 2.2, and finally check the directed percolation hypothesis and analyze the results in Section 2.3.

1. Definitions and notations

In this section we recall the definition of a CA to fix notations, then define the perturbed CA.

1.1. Usual definition of CA

Definition 1.1. A Cellular Automaton (CA) is a tuple (Q, d, V, δ) where

- Q is the finite set of *states*;
- $d \in \mathbb{N}^*$ is the *dimension*;
- $V = \{v_1, \dots, v_{|V|}\}$, the *neighborhood*, is a finite set of vectors in $\mathbb{Z}^d$;
- $\delta : Q^{|V|} \to Q$ is the *transition rule*;

The *cell space* is $\mathcal{U} := \mathbb{Z}^d$. A *configuration* sets the state of each cell: it is a function $c : \mathcal{U} \to Q$.

Here is the dynamic: given a configuration c, the next configuration c' is obtained by updating all sites at once by applying δ: $c'(z) := \delta\big(c(z+v_1), \dots, c(z+v_{|V|})\big)$. We extend the notation δ by defining $\delta(c) := c'$.

1.2. The Forcing Model

Given a CA, we consider two initial configurations c_0^1 and c_0^2. Those configurations are random, the state of each cell is drawn independently from the the others, with all states of Q equiprobable. At each time step, each configuration is updated according to δ, then a stochastic synchronization step F_q between both configurations occurs. The event studied is whether both configuration eventually become identical (they are said to have *synchronized*).

F_q consists in, for each cell z independently, doing nothing with probability q, and forcing both configurations to have the same state with probability $1-q$. When we force both configurations to agree on cell z, the state is chosen randomly uniformly between $c^1(z)$ and $c^2(z)$:

$$\text{For each cell } z \text{ independently,} \quad F_q(c^1(z), c^2(z)) := \begin{cases} c^1(z),\, c^2(z) & \text{with probability } q \\ c^1(z),\, c^1(z) & \text{with probability } \frac{1-q}{2} \\ c^2(z),\, c^2(z) & \text{with probability } \frac{1-q}{2} \end{cases}$$

Combining F_q and δ, we get:

$$c_{t+1}^i(z) := \begin{cases} \delta(c_t^i)(z) & \text{with probability } q & (a) \\ \delta(c_t^1)(z) & \text{with probability } \frac{1-q}{2} & (b) \\ \delta(c_t^2)(z) & \text{with probability } \frac{1-q}{2} & (c) \end{cases}$$

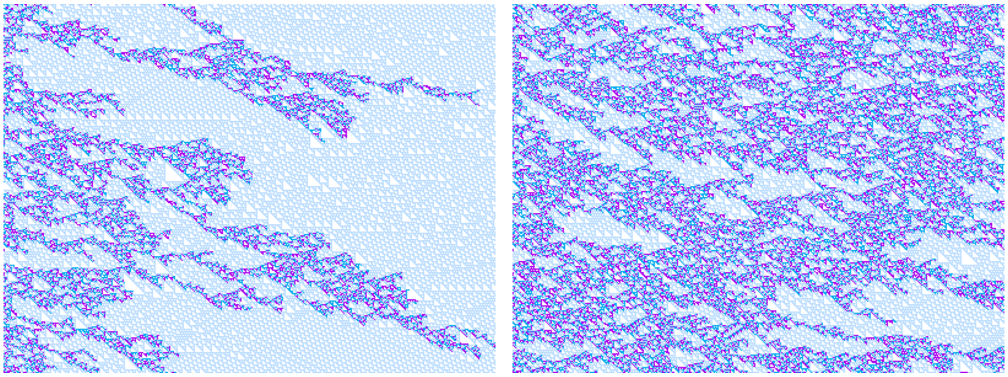
For each cell z independently, the same choice among (a), (b) and (c) is made for both configurations. An example is given on Figure 1.

The probability q is a parameter of the model. The case $q = 1$ corresponds to the unperturbed CA, or two independently evolving configurations, while the case $q = 0$ imply total synchronization just after the first step.

Note that if both states are equal, the forcing has no effect. Since the decision of forcing or not is independent for each cell, we can equivalently say that we try to force only if both cells are different. This is the presentation chosen in [10].

Proposition 1.2 (the case of strong coupling). *On finite configurations of size n, if $q \leqslant \frac{1}{|V|}$ then synchronization occurs in $O(n)$ expected time.*

Proof. The density of disagreement cells is, on average, multiplied by q each time F_q is applied. It is multiplied by at most $|V|$ when applying δ, since a disagreement cell can make only its neighbors become disagreement cells. If $|V|q < 1$, the expectancy of this density is thus exponentially decreasing. ■



Time goes from left to right. Configurations c^1 and c^2 are superimposed. Agreeing cells (i.e. cells having the same state in both configuration) are plotted light, disagreeing cells are plotted dark. One can see typical patterns of the original rule 110 in the light zones.

Figure 1: Forcing model applied on rule 110.

The bound on q of proposition 1.2 is loose:

- not many rules make disagreement spread at the speed of light in all configurations,
- disagreement sites are not isolated, precisely because they spread.

Link to Another Model. In [13], the authors study on finite configurations what they call self-synchronization, i.e. they have only one configuration that they try to "synchronize" with itself, which means reaching a stable configuration. They do it by setting $c_t^2 := c_{t-1}^1$, i.e. synchronizing the configuration with the configuration at the previous time step. This means that at each time step t, for each cell independently, the cell is reset to its state at step $t-1$ with probability $\frac{1-q}{2}$. This is equivalent to updating the cell with probability $\frac{q+1}{2}$ and doing nothing otherwise. The latter model, which updates only some cells at each time step, has been studied extensively both experimentally and analytically in [4].

2. Experimental study

In this section, we systematically study (in the forcing model) the ECA, that is, CA with $Q = \{0, 1\}$, $d = 1$, $V = \{-1, 0, 1\}$. We show that there are only 2 possible behaviors: synchronization even with the slightest forcing strength, and phase transition when q varies.

2.1. Classification of ECA in the forcing model

Here is the protocol of our experiments. We call *run* the temporal evolution of a CA when all parameters (rule, size n, probability q and two initial configurations) are chosen. We stop the run when both configurations are synchronized, or when a predefined maximum running time has been reached.

Let us describe the parameters we used.

(1) Number of cells n. The main points when choosing n is to check that the results do not depend on a particular choice of n, in particular that n is big enough. Some authors (like [14]) suggest that small is enough ($n = 30$), others (like [3]) state the opposite, and we follow the latter. A similar problem studied in [4] shows a stable behavior for $n \geqslant 200$. We set $n = 2\,000$ and check that the results do not change for $n = 500$.
(2) Number of computation steps. To measure the asymptotic density of disagreeing cells ρ, we let the automaton run for 200 000 steps, then measure the density averaged over 10 000 steps.

    According to directed percolation theory, near a phase transition, the automaton can take an arbitrarily long time before settling down to the asymptotic density. So, for a few choices of q, those parameters are not sufficient to measure the true asymptotic density. However, they are big enough to *detect* that there is a transition point, and then study more precisely what happens there.
(3) q. We try to sample the entire range. For each of the 88 rules, we do 999 runs: one for each value of q ranging from 0.001 to 0.999.

    One might want to average over many runs. To show that we do not need to, we plot ρ versus q. The smoothness of the resulting curve (Figure 2) shows that the variance between runs is low.

    The random seed for deciding which cells to update at each step is distinct for each value of q.
(4) The initial configurations are random (each cell is in state 0 with probability 0.5, independently from the other cells) and distinct for each value of q.

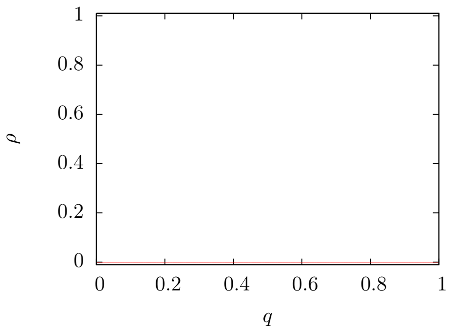
When we apply this protocol to all ECA, there are only two different situations occuring. A typical plot of each case is on Figure 2.

- 68 ECA have a trivial behavior: both configurations always synchronize within the given time, for all values of q.
- 20 ECA exhibit a phase transition. For some q_c (depending on the rule):
  - If $q < q_c$, both configurations rapidly synchronize.
  - If $q > q_c$, the density of disagreeing cells settles to non zero value for a long time.

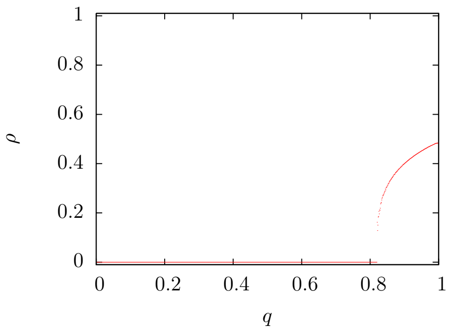
We deal with finite configurations and can thus be subject to finite size effects. One effect of notable importance is the following. Take a rule with a phase transition, there is an update rate q for which the rule is synchronizing, i.e. the pair of configurations reaches total agreement in polynomial time. In the non synchronizing regime, with low probability, the outcome of the random bits determining which cells to force can make the CA simulate the synchronizing regime for a fixed number of steps. So, if a CA can synchronize for a given q, it can synchronize for any $0 < q < 1$. The true asymptotic regime is thus always synchronization.

In other words, when the density settles to non zero value (case $q > q_c$), it still fluctuates randomly around this value. Fluctuations eventually make the density touch 0, which is asborbing.

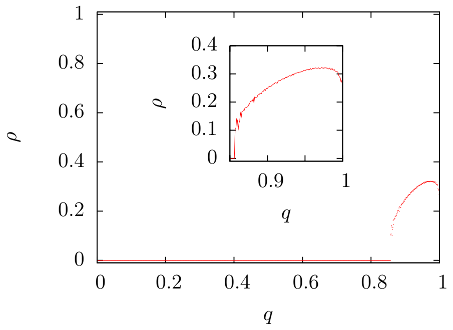
However, these fluctuations become smaller as n increases, and the limit when $n \to \infty$ should be that the density does not reach 0 anymore. Moreover, this first experiment is used to detect rules that have an interesting behaviour, and all rules of the second class will be checked in detail in Section 2.3.



A rule for which both configurations always synchronize (26)



A rule with phase transition (110)



Rule 57

Figure 2: Asymptotic density of disagreeing cells ρ versus q.

2.2. Phase transition and directed percolation

This section recalls the minimal background about directed percolation.

2.2.1. *Phase transition.* A phase transition is an abrupt change in macroscopic properties of a system with only a small change of a control parameter, say T, around a critical value T_c. This paper is concerned only with second order phase transitions, or continuous phase transitions, which can be characterized by *critical exponents.* If one let the parameter T vary near the phase transition (occuring at $T = T_c$), all other variables being fixed, a measurable quantity C has a power law behavior $C \propto |T - T_c|^\beta$ at least on one side of T_c. Several exponents are defined, depending on the quantity measured.

Remarkably, many systems with no *a priori* relation turn out to have the same critical exponents. A *universality class* is defined as all the systems having the same set of critical exponents.

2.2.2. *A conjecture on damage spreading.* Chaos theory deals with the sensitivity to initial condition of *deterministic* systems. To also study the influence of small perturbations on stochastic systems, the authors of [8] introduced *damage spreading.* In this model, two copies of a stochastic model are run in parallel with the same source of random bits, starting from different initial configurations (often they are set to differ in exactly one site).

One measures the temporal evolution of the proportion of differing sites, called the Hamming distance. If this goes to zero, i.e. if both copies become identical, the initial "damage" has "healed", otherwise the damage is said to spread.

There is a conjecture by [5] stating that, if a transition occurs between healing and spreading in a stochastic spin model, the universality class of this phase transition is always the same, namely the one of *directed percolation*, which is presented in the next paragraph.

There are some conditions for this conjecture:

(1) Only short range interactions in time and space,
(2) translational invariance,
(3) non vanishing probability for a site to become healed locally,
(4) the transition does not coincide with another phase transition.

Point 2 is easily fulfilled for CA. Point 3 is a direct consequence of the definition of the forcing model. About point 4 (no simultaneous transition), we know of no other transition. We discuss point 1 in Section 2.3.3.

2.2.3. *The Model of Directed Percolation.* A more detailed introduction to directed percolation can be found in [4] (note that this paper cites a different conjecture of Grassberger than the one we deal with). A survey of directed percolation is contained in [7], which also covers *damage spreading.*

Isotropic percolation was first defined when studying propagation of a fluid through a porous medium. It has been mathematically modelled as an infinite square grid where each site has the four nearest sites as neighbors. Each bond between two neighbors can be open (letting the fluid go through) with probability p or closed with probability $1-p$, independently of all other bonds.

The question is whether the fluid inserted at one point will pass through the medium, i.e. whether this point is part of an infinite network of sites connected by open bonds.

Directed percolation appears when one adds gravity to the model, i.e. when the fluid is only allowed to travel in one direction (Figure 3). Static 2D directed percolation can also be seen as a 1D dynamical model where some sites are "active" (where active can mean wet, infected, etc.). An active cell can stay active or die (become inactive), and make its neighbors active. Depending on the probabilities of these possibilities, active regions spread or disappear. Cells can only have an influence on the future states of their neighbors, thus the *directed* percolation.

The macroscopic quantity measured is the density of active states as a function of p and time, $\rho(p,t)$. It is zero in one phase and non-zero in the other. There exists a critical probability p_c which is the limit between two phases:

- For $p < p_c$, the asymptotic density $\rho(p,\infty)$ is 0;
- for $p > p_c$ we have a power law $\rho(p,\infty) \propto (p-p_c)^\beta$;
- for $p = p_c$ the density goes to 0 as $\rho(p_c,t) \propto t^{-\delta}$.

2.3. Directed percolation in the forcing model

In our model, the active sites are the cells where the configurations disagree. Asymptotic density of such sites is written $\rho(q)$. The pairs of configurations where all cells agree

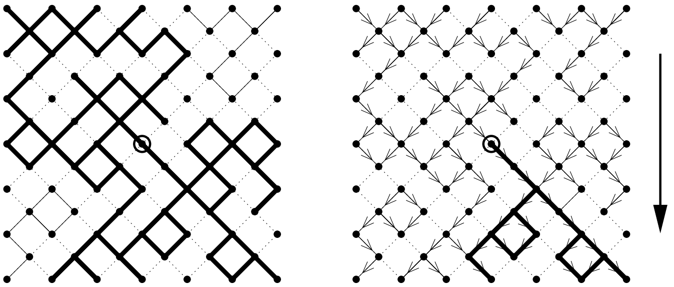


Figure 3: Isotropic (left) and directed (right) percolation. Figure reprinted from [7].

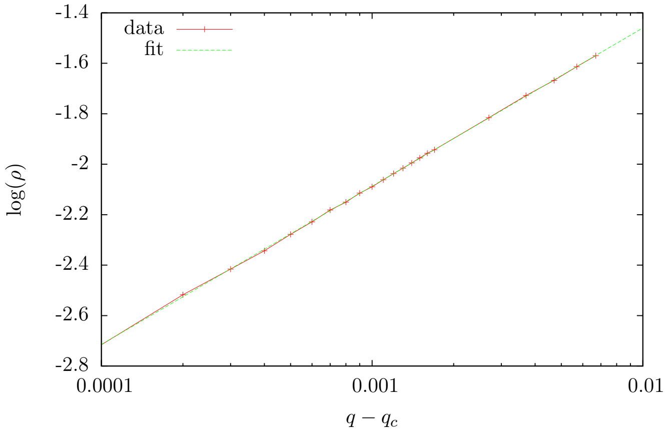


Figure 4: Measuring β for rule 110.

constitute the absorbing set. Percolation transition (synchronization or not) appears when varying q, see Figure 1. (Note that there is no direct relation between q and p, because of the underlying CA dynamic.) The aim is thus to identify β assuming that

$$\rho(q) \propto |q - q_c|^\beta \tag{2.1}$$

for some q_c. Like many authors [4, 5], we will focus on β and consider it as sufficient to test directed percolation.

2.3.1. *Measure of β.* Two methods to measure β were compared in [12], we use the following one. We plot $\log \rho(q)$ versus $\log(q - q_c)$ for values of q near q_c and adjust q_c to get a straight line. Once q_c is fixed, we fit a straight line, the slope of which is an estimator of β. See Figure 4 and Equation 2.1. It is important to do the fit against $\log \rho$ (and not ρ), so that all errors get the same weight when fitting a line on the log-log plot.

The protocol is only semi-automatic. We try increasing values of n between 10 000 and 1 000 000 to get a reasonably smooth line on the density versus time plot. All other parameters being set, different values of n give fluctuations around the *same* asymptotic

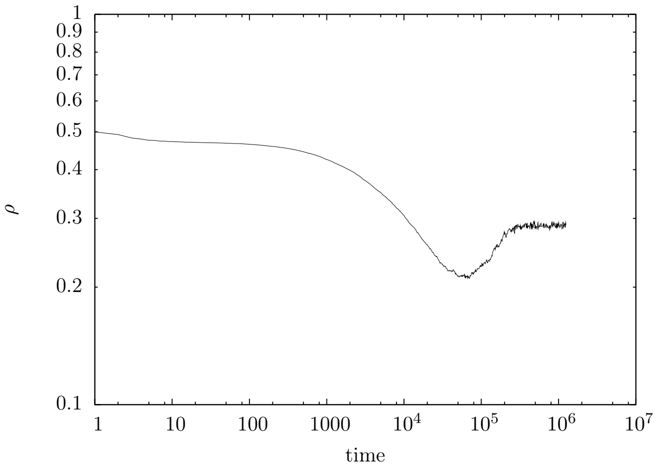


Figure 5: ρ versus time for rule 58 and $q = 0.9999$ (log log scale).

density, and larger n yields smaller fluctuations and thus greater precision. Also, this allows us to check that the number of steps required for the density versus time plot to become horizontal is not affected by n.

We then visually check that the density has reached a steady state, then average the density over at least half a decade. This yields one measure point. We repeat this process for several values of q near q_c.

Note the misleading diagram of rule 58 (Figure 5): running the experiment only for 50 000 time steps would lead one to conclude that the density will reach zero.

2.3.2. *Results.* The fit gives the ranges of Table 1, taking into account uncertainty about q_c and which points to keep for the fit. Experimental value for β measured on other systems is 0.276. As expected, all models undergoing a phase transition (that is, in our case, all models not always synchronizing) seem to belong to the universality class of directed percolation.

Rule 110 has been measured with higher precision than the other rules to check the influence of particles, discussed in Section 2.3.3.

Thresholds for rules 58 and 62 are very close to 1 and too sparse sampling of the values of q could miss their transition. For those rules, a tiny synchronization strength (roughly one percent) is already enough for both configurations to synchronize, but there is still a strength at which they do not synchronize.

A few rules have very close thresholds and this is no coincidence: there is a way to relate their dynamics.

- 18 and 146. The only difference between those rules is on the local configuration 111: $\delta(1,1,1) = 0$ for 18 and $\delta(1,1,1) = 1$ for 146. But the only way to have a pattern 1^k (with $k \geqslant 3$) under the dynamic of 146 is to have the pattern 1^{k+2} on the previous configuration. So, provided there is at least one 0 in the initial configuration, such patterns rapidly disappear and the dynamics of 18 and 146 are then identical.

rule	q_c	β
9	0.9630(2)	0.306 ±0.031
18	0.8092(2)	0.285 ±0.021
22	0.7727(2)	0.268 ±0.018
25	0.9570(2)	0.288 ±0.016
30	0.7935(1)	0.269 ±0.016
41	0.7954(1)	0.277 ±0.013
45	0.7946(1)	0.275 ±0.011
54	0.8387(2)	0.283 ±0.012
57	0.8546(1)	0.295 ±0.026
58	0.9968(2)	0.26 ±0.03
60	0.8094(1)	0.27 ±0.015
62	0.9854(2)	0.291 ±0.027
90	0.8094(2)	0.263 ±0.022
105	0.6789(2)	0.268 ±0.01
106	0.8498(1)	0.275 ±0.009
110	0.81930(1)	0.272 ±0.005
122	0.7850(1)	0.274 ±0.011
126	0.7892(2)	0.269 ±0.011
146	0.8094(2)	0.259 ±0.021
150	0.6789(2)	0.265 ±0.013

Numbers in parenthesis give the precision:
"0.9630(2)" means $q \in [0.9628;\ 0.9632]$.

Table 1: q_c and β for all ECA undergoing a phase transition.

- 60 and 90. 90 means "xor between my left and my right neighbors". 60 means "xor between me and my left neighbor". Let us consider a space-time diagram $\{c_t(z) \mid t \in \mathbb{N},\ z \in \mathcal{U}\}$ obeying rule 90:
$$\forall z \quad \forall t \qquad c_{t+1}(z) \;=\; \delta(c_t(z-1),\ c_t(z),\ c_t(z+1)) \;=\; c_t(z-1) \oplus c_t(z+1)$$
If we extract the space-time diagram $\{c'_t(z) := c_t(2z-t) \mid t \in \mathbb{N},\ z \in \mathcal{U}\}$, it obeys rule 60:
$$\begin{aligned}\forall z \quad \forall t \qquad c'_{t+1}(z) &= c_{t+1}(2z-t-1)\\ &= c_t(2z-t-2) \oplus c_t(2z-t)\\ &= c'_t(z-1) \oplus c'_t(z)\end{aligned}$$
Thus, 90 simulates two half-size configurations of 60.
- 105 and 150. 150 means "xor of the states of the neighbors" while 105 means "compute the output of 150 and take the opposite state". Thus, the agreement status of on cell only depends on the agreement status of its neighbors in the previous configurations (not on its actual state in both configurations). Which allows us to conclude that, if we run both rules on the same pair of initial configurations, exactly the same cells at each time step will disagree.

2.3.3. *The case of rule* 57. In previous papers, only chaotic rules have been the subject of interest for studying synchronization and phase transition in the forcing model. But let us consider rule 57 which, when unperturbed, converges quickly to a period 2 orbit (in this case a checkerboard). It is thus not chaotic. Nonetheless, if one adds forcing, a phase transition occurs. This shows that this phase transition does not require the CA to be chaotic. However, all ECA in class 3 or 4 of Wolfram classification (i.e. "chaotic" or "complex" ECA) undergo a phase transition.

This rule is of further interest because it has particles (in the forcing model) and thus long range correlations, as seen on Figure 6. Rule 110 is also known to have particles (see Figure 1) and thus exactly the same kind of long range correlations. We have shown that both rules undergo a phase transition of the directed percolation class. This shows that point 1 of Grassberger's conjecture (page 256), while useful for discarding models that do not belong to this class, might be too restrictive.

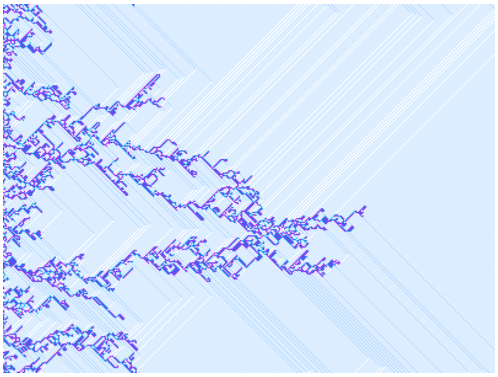


Figure 6: A Space-time diagram of rule 57 in the forcing model.

The last insight given by this rule is the following. On Figure 2.c, one sees that for q between 0.98 and 0.999, more forcing (q closer to 0) means higher asymptotic density. In other words, increasing the number of cells we force to be equal at each time step makes the density of disagreeing cells higher. Note that applying the forcing step F_q is equivalent to applying two forcing steps $F_{\sqrt{q}}$. In this case, applying the second one would raise ρ.

3. Conclusion and perspectives

We have studied a way to perturb a cellular automaton, with a perturbation related to the original rule. For 68 ECA rules, the slightest perturbation allows to synchronize two random initial configurations. For the 20 remaining ones, there is a threshold on the couplin strength.

When studying the behavior close to the threshold, we confirm the results of the first experiment and show that there is a phase transition belonging to the universality class of directed percolation.

Experiments have to be run on finite configurations, but we checked that the number of cells n does not influence the outcome, increasing n only increases accuracy. There is evidence that the only finite size effect (the fact that zero density is absorbing) occurs only after an exponential time, and thus does no harm here.

Therefore, we expect the theoretical model to behave like the experiments, i.e. to undergo a phase transition with an exponent of $\beta \simeq 0.276$ (note that about the 1D directed percolation, the literature have precise experimental values for β but no analytical derivation).

There are certainly some rules that can be analytically studied and be proven of undergoing a phase transition. Simulation between directed percolation models is a strategy for this kind of result.

Other open questions remain. An obvious generalization is to test in which proportion this phenomenon occurs in CA with more states, more dimensions or more neighbors.

Alos, in this model, there is a symmetry between both configurations (they are treated equally). In a context were the aim is to control the system, it would natural to study a "master/slave" setting: when the random outcome tells to make two cells equal, the state would always be copied from the first configuration to the second.

Finally, one could imagine relevant ways of coupling more than two instances of the CA. But it would be even more interesting to mix this coupling scheme with the coupling studied in [12] and see how their respective effects combine.

Links. Space-time diagrams for all ECA can be found on `http://www.rouquier.org/jb/recherche/eca`. Source code used for the simulations is at `http://cimula.sf.net`.

References

[1] Vassil N. Alexandrov, G. Dick van Albada, Peter M. A. Sloot, and Jack Dongarra, editors. *Computational Science - ICCS 2006, 6th International Conference, Reading, UK, May 28-31, 2006, Proceedings, Part III*, volume 3993 of *Lecture Notes in Computer Science*. Springer, 2006.

[2] Franco Bagnoli and Fabio Cecconi. Synchronization of non-chaotic dynamical systems. *Physics Letters A*, 282:9–17, April 2001.

[3] Simon R. Broadbent and John M. Hammersley. Percolation processes in crystals and mazes. In *Proceedings of the Cambridge philosophical society*, pages 629–641, 1957.

[4] Nazim A. Fatès. Asynchronism induces second order phase transitions in elementary cellular automata. *Journal of Cellular Automata*, march 2007.

[5] Peter Grassberger. Are damage spreading transitions generically in the universality class of directed percolation? *J Stat Phys*, 79:13–23, September 1995.

[6] Peter Grassberger. Synchronization of coupled systems with spatiotemporal chaos. *Phys. Rev. E*, 59(3):R2520–R2522, Mar 1999.

[7] Haye Hinrichsen. Nonequilibrium critical phenomena and phase transitions into absorbing states. *Advances in Physics*, 7:815–958, Nov 2000.

[8] Stuart A. Kauffman. Emergent properties in random complex automata. *Physica D Nonlinear Phenomena*, 10:145–156, Jan 1984.

[9] Amos Maritan and Jayanth R. Banavar. Chaos, noise, and synchronization. *Physical Review Letters*, 72:1451–1454, March 1994.

[10] Luis G. Morelli and Damián H. Zanette. Synchronization of stochastically coupled cellular automata. *Phys. Rev. E*, 58(1):R8, Jul 1998.

[11] Arkady S. Pikovskii. Synchronization and stochastization of array of self-excited oscillators by external noise. *Radiophysics and Quantum Electronics*, 27:390–395, May 1984.

[12] Jean-Baptiste Rouquier and Michel Morvan. Coalescing cellular automata. In [1], pages 321–328, 2006.

[13] Juan R. Sánchez and Ricardo López-Ruiz. Self-synchronization of cellular automata: An attempt to control patterns. In [1], pages 353–359, 2006.

[14] Andrew Wuensche. Classifying cellular automata automatically: finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter. *Complexity*, 4(3):47–66, 1999.

[15] Damián H. Zanette and Luis G. Morelli. Synchronization of coupled extended dynamical systems: a short review. *International Journal of Bifurcation and Chaos*, 13(4):781–796, 2003.

Journées Automates Cellulaires 2008 (Uzès), pp. 262-262

OPTIMAL TIME SELF-ASSEMBLY FOR SQUARES AND CUBES

FLORENT BECKER [1], ÉRIC RÉMILA [1], AND NICOLAS SCHABANEL [2]

E-mail address: florent.becker@ens-lyon.fr

[1] LIP, ÉNS Lyon

[2] DIM, Universidad de Chile

Abstract. Self-assembly is the process by which small entities combine themselves into a bigger shape by local interactions in such a way that the resulting aggregates have interesting global properties. In particular, self-assembling tile systems are a model for assembling DNA-based nano artefacts which resembles cellular automata. They consist of Wang tiles with glues on their sides. These glues have different strength, and a minimal total strength is required to add a tile to a pattern. Thus, the tiling algorithm is embedded into the Wang tiles themselves.

In the currently known constructions, most of the effort is put on guaranteeing the size of the output object, whereas the geometrical efficiency of the assembly of the shape itself is left aside. In this talk, we will present a framework to obtain provably time efficient self-assembling tile systems. Our approach consists in studying how the flow of information has to circulate within the desired shape to guarantee an optimal time construction. Instead of starting with a tileset and studying its time performances, we go the other way around: we start from the trace of an optimal run for assembling a shape and transform it into a tileset with that behaviour. This study can yield an adequate ordering of the tiling process from which one can deduce a provably efficient tile system for that shape.

In particular, in 3 dimensions, proving the correctness, let alone the time optimality of a construction is a tedious process. With our method, this can almost be reduced to a few elementary properties of a well chosen order.

We will present this approach on squares and cubes, for which we obtain time optimal self-assembling tile systems.

Journées Automates Cellulaires 2008 (Uzès), pp. 263-274

COMPUTATIONAL MODELS AND CODES ON GRAPHS (EXTENDED ABSTRACT)

H. BEN-AZZA [1] AND W. BAOUSAR [2]

[1] ENSAM, BP 4024, Bni Mhammed, Meknès, Morocco
E-mail address: hbenazza@yahoo.com

[2] Faculté des Sciences et Techniques, Fès, Morocco

Abstract. In this paper, we summarize some results concerning two fields: (1) models of computation including Boolean networks and cellular automata; (2) error correcting codes, whose purpose is to communicate over a noisy channel. We consider the class of codes on graphs as exemplified by low-density parity and expander codes. For this class we also give their linear programming formulation, introduced by Feldman et al. and we answer a question of Feldman by providing a linear programming formulation for codes over non binary alphabets.
The two fields are enriching each other, at least for studying the complexity of error correcting codes.

Introduction

The 'natural' idea of using error correcting codes in the study of cellular automata has been pointed out in [22, 14]. This is our first motivation.

The class of low-density parity-check (LDPC) codes, and its belief propagation decoding [15, 27, 8] has motivated researchers to understand the excellent performance of these codes. A novel approach [11], by the use of linear programming (LP), gives many insights into iterative decoding of codes on graphs. A second motivation of this paper, is to formulate a LP decoder for non binary error correcting codes. Below, in subsection 4.2.4, we give an answer to this question, which also inherits some properties (like maximum likelihood certificate) from the binary case.

A third motivation is to understand, a new flipping algorithm [26, 2, 31], with the use of the chromatic number of an expander graph (see subsection 4.3). But, we only sketch the decoder.

Section 1 defines Boolean networks (or circuits) which consists of a directed acyclic graph labeled by inputs and gates (Boolean functions from a complete basis) (see [10, 22, 3]). The concept of *uniformity* (answering the question of how to construct a family of circuits) introduced in [4], and systematically studied in [24], is reformulated in this section for the

Key words and phrases: Cellular Automata, Computational Complexity, Error Correcting Codes, Boolean Networks.

needs of simulations between cellular automata and circuits. Section 2 introduces one-dimensional cellular automata (CA) and main complexity measures attached to them (see [9, 20, 22]). Next, in section 3 we summarize some results of simulations between the two models and the proofs are given in [3]. We say that a circuit is synchronous if all the paths from inputs to a given gate are of the same length. It is, technically easier to simulate by CA a synchronous family of circuits. The simulation of CA by circuits gives rise to uniform families of circuits in a strong sense. The notion of error correcting code [1, 19, 23] is briefly introduced in Section 4. It is a pleasant result [26] that the complexity of the flipping algorithm for expander codes is in the class NC. Section 5 is devoted to some questions implied by this short overview.

For notions on complexity theory and notations, we refer to [16, 29]; we denote, for example, by $DTIME(T(n))$ the class of languages recognizable in time $\mathcal{O}(T(n))$ by a deterministic Turing machine. The notation $DTIME, SPACE(T(n), S(n))$ stands for the (simultaneous) class of languages accepted by deterministic Turing machines working both in time $\mathcal{O}((T(n))$ and uses $\mathcal{O}((S(n))$ space. The same notion applies to alternating Turing machines (by the notation $ATIME, SPACE(T(n), S(n)))$.

1. Boolean networks

Let $G = (V, E)$ be a directed graph for which the set of vertices is $VG = V$ and the set of edges is $EG = E$. For a vertex $v \in VG$, we denote $\Gamma_i(v) = \{w : (w, v) \in EG\}$, and $\Gamma_o(v) = \{w : (v, w) \in EG\}$. $\Phi_i(v) = |\Gamma_i(v)|$ (fan-in), $\Phi_o(v) = |\Gamma_o(v)|$ (fan-out). The induced graph on a set $X \subseteq VG$, is denoted by $G[X]$. $I(G) = \{v : \Phi_i(v) = 0\}$ (input of the graph). A path is a sequence of vertices $P = (v_0, ..., v_k)$ such that $(v_i, v_{i+1}) \in E$, for $i = 0, ..., k-1$; the origin of P is v_0 and its inverse is $(v_k, ..., v_0)$. Let $\beta_n = \{f \mid f : \{0,1\}^n \to \{0,1\}\}$ denote the set of Boolean functions on n arguments $X_n = \{x_1, ..., x_n\} \in \{0,1\}^n$. A subset $\Omega \subseteq \beta_2$ is a complete basis if any function of β_n can be expressed by composing elements from Ω.

Definition 1.1. Let $\Omega \subseteq \beta_2$. An $\Omega-$circuit or $\Omega-$ Boolean network B_n is a pair (G, op) consisting of a directed acyclic graph $G = (V, E)$ and a labeling map $op : VG \to \Omega \bigcup X_n \bigcup \{0, 1\}$ such that

- **(i):** for each vertex v with $\Phi_i(v) = 0$, either $op(v) \in X_n$ (input variable), or $op(v)$ is the constant gate 0 or 1.
- **(ii):** if $op(v) \in \Omega$, then $\Phi_i(v)$ is equal to the rank of the operation $op(v)$ and element of $\Gamma_i(v)$ corresponds uniquely to an argument of $op(v)$. The vertex v is called a gate and there exists a unique gate with fan-out 0 called the output gate.
- **(iii):** furthermore, if the induced graph $G[VG - I(G)]$ is a tree, then the circuit is called a formula.

The class of $\Omega-$circuits with n inputs is denoted by $\mathcal{B}_n(\Omega)$. In a natural way, we associate to each vertex v of a circuit B_n a Boolean function $res(v)$, defined by induction. We say that B_n, with output gate v, computes a function $f \in \beta_n$ if $res(v)(X_n) = f(X_n)$ for all $X_n \in \{0,1\}^n$. The family $\{B_n : B_n \in \mathcal{B}_n(\Omega)\}$ computes the function $f : \{0,1\}^* \to \{0,1\}$, if for each n, B_n computes the function f restricted to $\{0,1\}^n$.

Definition 1.2. Let $T \in \mathcal{B}_n(\Omega)$.

- **(i):** The *complexity* of T is $C_\Omega(T) = \mid VT \mid -n$.
- **(ii):** The depth of T is $C_\Omega(T) = max\{\mid P \mid : P \text{ is a path in } T\}$.

Definition 1.3. Let $f \in \beta_n$.

(i): The *complexity* of f is $C_\Omega(f) = min\{C_\Omega(T) : T \in \mathcal{B}_n(\Omega)$ computes $f\}$.

(ii): The *depth* of f is $D_\Omega(f) = min\{D_\Omega(T) : T \in \mathcal{B}_n(\Omega)$ computes $f\}$.

(iii): The *formula size* of f is $\mathcal{L}_\Omega(f) = min\{C_\Omega(T) : T \in \mathcal{B}_n(\Omega)$ is a formula computing $f\}$.

Let $Z, T : \mathbb{N} \to \mathbb{R}^+$. We define the complexity classes

$$SIZE(Z) = \{A \subseteq \{0,1\}^* \mid \exists\{B_n\} \text{ computing } A \text{ and } C(B_n) = \mathcal{O}(Z(n))\},$$
$$DEPTH(T) = \{A \subseteq \{0,1\}^* \mid \exists\{B_n\} \text{ computing } A \text{ and } D(B_n) = \mathcal{O}(T(n))\}.$$

The notion of the *level* of a vertex v in $T \in \mathcal{B}_n$ is recursively defined by :

$$N(v) = \begin{cases} 0 & \text{if } \Phi_i(v) = 0, \\ max\{N(u) : u \in \Gamma_i(v)\} + 1\} & else. \end{cases}$$

The set of vertices of level i is $N_i = \{v \in T : N(v) = i\}$, and this induces a partition $VT = \bigcup_{i=0}^{D(T)} N_i$. We denote

$$L(T) = max\{|N_i| : 0 \le i \le D(T)\} \text{ and } L_1(T) = max\{|N_i| : 0 < i \le D(T)\}.$$

Definition 1.4. (i)The *width* of $T \in \mathcal{B}_n$ is

$$W(T) = \max_{0 \le i \le D(T)} \mid \{ v \in VT : N(v) \le i \text{ and } \exists w \in VT \text{ such that } N(w) > i \text{ and } (v,w) \in ET\} \mid$$

(ii)Let $Z : \mathbb{N} \to \mathbb{R}^+$. We define the complexity class

$$WIDTH(Z) = \{A \subseteq \{0,1\}^* \mid \exists\{B_n\} \text{ computing } A \text{ and } W(B_n) = \mathcal{O}(Z(n))\}$$

Let $B_n \in \mathcal{B}_n$. Its standard encoding $\overline{B_n}$ is the set of tuples $< v, g, l, r >$ with the following meaning: v is a vertex of B_n, $g = op(v)$ the type of v (variable, 'and', 'or' gates , etc), and l, r are the ordered set of arguments of v. For a family $\mathcal{B} = \{B_n\}$ of circuits, its extended language L_{EC} is the set of tuples $< n, v, P, y >$ where

1): the sequence P is the inverse of a path in B_n with $\mid P \mid \le \log C(B_n)$;

2): if P is the empty sequence, then y is $op(v)$, else y is the origin of the inverse of P.

The direct language L_{DC} is the same as the extended language with the restriction that $\mid P \mid \le 1$. We note that $\mid \overline{B_n} \mid = \mathcal{O}(C(B_n) \log C(B_n))$.

It is important to construct families of circuits, and the following notions of *uniformity* capture this notion:

Definition 1.5. Let $\mathcal{B} = \{B_n\}$ be a family of circuits of size $Z(n)$ and depth $T(n)$. The family $\mathcal{B}$ is said

(i): U_D-uniform if $L_{DC} \in DTIME(\log Z(n))$;

(ii): U_E-uniform if $L_{EC} \in DTIME(\log Z(n))$;

(iii): U_{E^*}-uniform if $L_{EC} \in ATIME, SPACE(T(n), \log Z(n))$;

(iv): U_{BC}-uniform if $(1^n \to \overline{B_n}) \in DSPACE(\log Z(n))$;

(v): U_B-uniform if $(1^n \to \overline{B_n}) \in DSPACE(T(n))$;

Let the symbol $X \in \{U_D, U_E, U_{E^*}, U_{BC}, U_B\}$, and let A, B symbols designating complexity measures on circuits (like $SIZE, DEPTH$). Then by $X - A(Z(n))$ we mean the class of languages accepted by X uniform family of circuits using $\mathcal{O}(Z(n))$ of the resource A. The simultaneous class $X - A, B(Z(n), T(n))$ denotes the class of languages L such there exists a family of circuits $\mathcal{B} = \{B_n\}$ accepting L with the conditions that $\mathcal{B}$ is X

uniform, and the resources A, B of B_n are upper bounded simultaneously by $\mathcal{O}(Z(n))$ and $\mathcal{O}(T(n))$, respectively. Note that this notation may be extended to more than two complexity measures for any model of computation, and is generally distinct from the intersection of complexity classes.

The condition (iii) in the above definition is relative to alternating Turing machines, and we note that the U_E-uniformity is the strongest condition. The class NC contains problems computable simultaneously in polynomial size hardware and poylogarithmic parallel time, and it is insensitive to uniformity conditions (for $k \geq 2$):

Definition 1.6. Let $k \geq 1$ be an integer.

(i): $NC^k = U_{BC} - SIZE, DEPTH(n^{\mathcal{O}(1)}, \log^k(n))$
(ii): $NC = \bigcup_{k\geq 1} NC^k$
(iii): $SC = DTIME, SPACE(n^{\mathcal{O}(1)}, \log^{\mathcal{O}(1)}(n))$

One precise exemplification of the so-called the thesis of parallel computation is [4]:

Theorem 1.7.

$$U_B - DEPTH(T) \subseteq DSPACE(T), NSPACE(T) \subseteq U_B - DEPTH(T^2).$$

By a probabilistic proof, we have the lower bound:

Theorem 1.8. *(Shannon, see* [10]*)For all $\epsilon > 0$, for n sufficiently large, for almost all $f \in \beta_n$,*

$$C(f) > (1-\epsilon)\frac{2^n}{n}$$

By a constructive proof, we have the upper bound:

Theorem 1.9. *(Lupanov, see* [10]*)For all $\epsilon > 0$, for n sufficiently large, for almost all $f \in \beta_n$,*

$$C(f) < (1+\epsilon)\frac{2^n}{n}$$

But, it is a difficult task to exhibit a 'natural' language or a Boolean function with a non-trivial lower bound on the size.

2. One-dimensional cellular automata

We are interested in resource bounded computations by cellular automata.

Definition 2.1. A *one-dimensional cellular automaton* (CA for short) $\mathcal{A}$ consists of a line of finite automata indexed by the integers $\mathbb{Z}$, $(A_i)_{i\in\mathbb{Z}}$. Each A_i, called a *cell*, is a triple (Q, d, q) where :

(i): Q is a finite set of states;
(ii): $d : Q^3 \to Q$ is the transition function;
(iii): q is the quiescent state such that $d(q, q, q) = q$.

A *configuration* of the CA $\mathcal{A}$ is any sequence $(s_i)_{i\in\mathbb{Z}} \in Q^{\mathbb{Z}}$.

For $i \in \mathbb{Z}, t \in \mathbb{N}$, let $S_t(i)$ denote the state of the i^{th} cell at time t. For $t > 0$, we have

$$S_t(i) = d(S_{t-1}(i-1), S_{t-1}(i), S_{t+1}).$$

The transition function d of the CA $\mathcal{A}$ induces a *global transition function* $\Delta_{\mathcal{A}} : Q^{\mathbb{Z}} \to Q^{\mathbb{Z}}$, and we define the t^{th} iteration of this operator, $\Delta^t_{\mathcal{A}}$, by : for each configuration C,

$$\Delta^0(C) = C, \text{ and for } t > 0, \Delta^{t+1}(C) = \Delta(\Delta^t).$$

The input to the CA at time $t = 0$ is a word $w = w_0 ... w_{n-1} \in Q^n$ such that

$$S_0(i) = \begin{cases} w_i & \text{for } i = 0, ..., n-1 \\ q & \text{for } i < 0 \text{ or } i \geq n. \end{cases}$$

The initial configuration is $I_{\mathcal{A}}(w) = (s_0(i))_{i \in \mathbb{Z}}$. We suppose that Q contains two special states q_a (*accept*) and q_r (*reject*). The decision on an input is made by the cell 0 such that if $s_0(t) = s \in \{q_a, q_r\}$, then for all $t' > t$, $s_0(t') = s$. We say that a word w is accepted (resp. rejected) by the CA $\mathcal{A}$ if and only if there exists a time t such that $\Delta^t(I_{\mathcal{A}}(w))(0) = q_a$(resp. q_r). The language associated to $\mathcal{A}$ is

$$L(\mathcal{A}) = \{w \in Q^* : \mathcal{A} \text{ accepts } w\}.$$

The running time on the input w is

$$CTIME_{\mathcal{A}}(w) = min\{t : \Delta^t(I_{\mathcal{A}}(w))(0) \in \{q_a, q_r\}\},$$

For $T : \mathbb{N} \to \mathbb{R}^+$, we define the complexity class of languages L on some finite alphabet by

$$CTIME(T(n)) = \{L : \exists CA \text{ accepting } L \text{ in time } \mathcal{O}(T(n))\}.$$

The set of cells participating in a computation on an input w is introduced by :

$$CLARGE_{\mathcal{A}}(w) = max(\mid w \mid, max_{i,j}\{\mid i - j \mid : \exists t, t', \Delta^t(I_{\mathcal{A}}(w))(i) \neq q, \Delta^{t'}(I_{\mathcal{A}}(w))(j) \neq q\})$$

and for a bound function $S : \mathbb{N} \to \mathbb{R}^+$, we define the class

$$CLARGE(S(n)) = \{L : \exists CA\ \mathcal{A} \text{ accepting } L, \forall w \in L, \mid w \mid = n, CLARGE_{\mathcal{A}}(w) = \mathcal{O}(S(n))\}.$$

Finally, we define the simultaneous class :
$CTIME, LARGE(T(n), S(n)) = \{L : \exists CA\ \mathcal{A}$ accepting $L, \forall w \in L,$
$CTIME_{\mathcal{A}}(w) = \mathcal{O}(T(n))$ and $CLARGE_{\mathcal{A}}(w) = \mathcal{O}(S(n))\}$.

The firing squad lemma (FSL) [20], which synchronize computations, is:

Theorem 2.2. *There exists a CA $\mathcal{A} = (Q, d, q)$ with three distinguished states $G, F,$ and \$ with the following property. If $S_0(0) = G, S_0(n+1) = \$,$ and $S_0(i) = q$ for $i = 1, ..., n$, then for all $n \geq 1$, for all $i = 0, ..., n+1$, and for all $t = 0, ..., 2n+1$, we have : $\Delta^t(G, q^{(n-1)}, \$)(i) \neq F$, and $\Delta^{2n+2}(G, q^{(n-1)}, \$)(i) = F$.*

We make heavy use of FSL to simulate circuits by CA in the following results. We note also that it is easy to extend the above concepts to dimensions greater than one.

3. Relations between circuits and cellular automata

Proposition 3.1. *If the function $T(n) \in DTIME(\log T(n))$, then*

$$CTIME(T(n)) \subseteq U_E - SIZE, DEPTH(T^2(n), T(n))$$

Corollary 3.2.

$$CTIME, LARGE(T(n), L(n)) \subseteq U_E - SIZE, DEPTH, WIDTH(T^2(n), T(n), L(n))$$

For cellular automata of dimension h, we have

Proposition 3.3. *If the function $T(n) \in DTIME(\log T(n))$, then*

$$CTIME^h(T(n)) \subseteq U_E - SIZE, DEPTH(T^{h+1}(n), T(n))$$

We say that a circuit T is synchronous if for all $v \in VT$, all the paths originating from the inputs of T and ending at v have the same length. Given a circuit B_n, we may construct a synchronous circuit, denoted B'_n, computing the same function. This transformation is denoted $B_n \to B'_n$, and given a standard description $\overline{B_n}$, we can give the description $\overline{B'_n}$ with the bounds :

Lemma 3.4. *For any family $\{B_n\}$ of circuits, we have*

(i): $(\overline{B_n} \to \overline{B'_n}) \in DSPACE_f(\log C(B_n))$.
(ii): $C(B'_n) = \mathcal{O}(C(B_n) + L_1(B_n) \times D^2(B_n))$.
(iii): $D(B'_n) = D(B_n)$ *and* $W(B'_n) = W(B_n)$.
(iv): *If* $S(n) \geq \log n$, *then* $(1^n \to \overline{B_n}) \in DSPACE_f(S(n)) \Rightarrow$ $(1^n \to \overline{B'_n}) \in DSPACE_f(S(n))$, *where* $DSPACE_f(S(n))$ *is the class of functions computable by deterministic Turing machines in* $\mathcal{O}(S(n))$ *space.*

For any complexity measure A on circuits, we denote by A' the synchronous version.

Proposition 3.5. *There exists a CA such that on the input $\overline{B_n}$ describing the standard code of a synchronous circuit B_n, outputs the same value in time $\mathcal{O}(\left|\overline{B_n}\right| \times C(B_n) \times D(B_n))$*

Definition 3.6. A family $\{B_n\}$ of circuits is U_C-uniform if $(1^n \to \overline{B_n}) \in CTIME(C(B_n))$

Corollary 3.7. *For each synchronous U_C-uniform family $\{B_n\}$ of circuits, there exists a simulating CA running in time $\mathcal{O}(C^2(B_n) \times \log C(B_n) \times D(B_n))$.*

Corollary 3.8. *If $T(n), L(n) \geq n$, then*

$U_C - SIZE', DEPTH', LARGE'(T, D, L) \subseteq CTIME, LARGE(DT^2 \log T, L(\log D + \log L))$

Observe that this corollary relates three simultaneous resources on synchronous circuits to two simultaneous resources on CA. Let F_n be a formulae, then we may found an encoding $\widetilde{F_n}$ of size $|\widetilde{F_n}| = \mathcal{O}(\sum \Phi_o(X_n) \log n)$.

Proposition 3.9. *There exists a CA $\mathcal{A}$ such that on input $\widetilde{F_n}$ simulates F_n with*

(i): $CTIME_{\mathcal{A}}(\widetilde{F_n}) = \mathcal{O}(D(F_n) \times \mathcal{L}(F_n))$
(ii): $CLARGE_{\mathcal{A}}(\widetilde{F_n}) = \mathcal{O}(\left|\widetilde{F_n}\right|)$

4. Error correcting codes

Let $\mathbb{F} = \mathbb{F}_q$ be the field with q elements. A code $\mathbb{C}$ over $\mathbb{F}$ of length n is a subset of $\mathbb{F}^n$, whose elements are called codewords. The Hamming distance over $\mathbb{F}^n$ is $d(x, y) = |\{i : x_i \neq y_i\}|$. The minimal distance of $\mathbb{C}$ is $d = d(\mathbb{C}) = \min\{d(c, y) : x, y \in \mathbb{C} \text{ and } x \neq y\}$ and the relative minimal distance is $\delta(\mathbb{C}) = d/n$. The rate is $R = R(\mathbb{F}) = n^{-1} \log_q(|\mathbb{F}|)$. A scalar product is defined by $x \cdot y = \sum_{i=1}^n x_i y_i$. The dual of the code $\mathbb{F}$ is $\mathbb{F}^\perp = \{x \in \mathbb{F}^n : x \cdot y = 0, \forall y \in \mathbb{F}\}$.

Furthermore, if $\mathbb{C} \subseteq \mathbb{F}^n$ is a $\mathbb{F}$-vetor space of dimension k and minimal distance d, then we say it is an $[n, k, d]$ linear code. In this case, the code is given by its generating matrix

G or by its control matrix $H : \mathbb{C} = \{ xG : x \in \mathbb{F}^k\} = \{x \in \mathbb{F}^n : Hx = 0\}$. We note that if $\mathbb{C}$ is an $[n, k, d]$ linear code, then its dual is an $[n, n-k, .]$ linear code and $(\mathbb{F}^{\perp})^{\perp} = \mathbb{F}$.

For us, a probabilistic noisy channel is seen as a bipartite graph over two copies of the alphabet $\mathbb{F}$ such that edges are labeled by the transition probabilities $Pr(a \mid b)$, for $a, b \in \mathbb{F}$ meaning the probability of receiving a given that b was sent. When a codeword $y = y_1...y_n$ (we suppose that the symbols y_i have uniform probability of appearance in y and the channel is memoryless) is sent over the channel, at reception we get the corrupted word $\widetilde{y} = \widetilde{y_1}...\widetilde{y_n}$. Let us summarize this situation by the notation $y \rightsquigarrow \widetilde{y}$.

Decoding by maximum likelihood (ML) reduces to the computation of $D^{ML}(\widetilde{y}) = argmax_{y\in\mathbb{C}}Pr(\widetilde{y} \mid y)$, for which the corresponding decision problem is known to be NP-complete [6, 28]. More generally, a decoder for $\mathbb{C}$ is any function $D : \mathbb{F}^n \to \mathbb{C}\bigcup\{?\}$. The symbol '?' means that the decoder D may not decides a decoding for some corrupted $\widetilde{y} \in \mathbb{F}^n$. In the situation $y \rightsquigarrow \widetilde{y}$, the decoding error probability on y is $Pr(D(\widetilde{y}) \neq y)$. Let us denote $p^D(\mathbb{C})$ the sum of all decoding error probabilities over $y \in \mathbb{C}$. Then the celebrated Shannon's noisy coding theorem [25] states that there exists a number C (the channel capacity), for large length n, and for any $\varepsilon > 0$, there exists a code $\mathbb{C}$ of rate $< C$ and an accompanying decoder D such that $p^D(\mathbb{C}) < \varepsilon$. Note that the proof of this theorem is by a probabilistic method.

4.1. Codes on graphs

The girth $g(G)$ of a simple undirected graph $G = (V, E)$ is the size of its smallest cycle. A graph is said sparse when the number of edges is linear in the number of vertices (i.e. $|E| = \mathcal{O}(|V|)$. A graph $G = (V, E)$ is said an (α, β)-expander if for every $S \subset V$ with $\mid S \mid\leq \alpha$, we have $\mid N(S) \mid\geq \beta \mid S \mid$.

4.2. Linear programming formulation

4.2.1. *From ML decoder to LP decoder.* Let $\mathbb{C}$ be a binary code, of length n, non necessary linear. We assume an arbitrary binary-input memoryless channel, and information words have equal probability.

Under these assumptions, the aim of ML decoder is given $\tilde{y}$ received, find y^* such that

$$y^* = \arg\max_{y\in\mathbb{C}} Pr[\tilde{y}\ received\ |y\ transmitted]$$

Let $\gamma_i = \ln \frac{Pr[\tilde{y}_i|y_i=0]}{Pr[\tilde{y}_i|y_i=1]}$ denote the log-likelihood ratio of a code bit y_i. We can show that y^* is an optimum of the problem

$$(ML)\begin{cases} \min \sum\limits_{i=1}^{n} \gamma_i y_i \\ subject\ to \\ y \in \mathbb{C} \end{cases}$$

Linear programming decoder is obtained by applying linear programming relaxation to the problem (ML): to every code bit we associate a variable f_i. We would like these variables to take on values in $\{0, 1\}$. But in relaxation f_i's will take on values in $[0, 1]$. We will define a polytope $\mathcal{P} \subseteq [0, 1]^n$ (set of linear constraints on the variables), by using the structure of the code, and the objective function will be $\min \sum\limits_{i=1}^{n} \gamma_i y_i$. The polytope $\mathcal{P}$ is said to be proper if we have

$$\mathcal{P} \bigcap \{0,1\}^n = \mathbb{C}.$$

It is easy to show that

$$\mathbb{C} \subseteq V(\mathcal{P}) \subseteq \mathcal{P} \subseteq [0,1]^n,$$

where $V(\mathcal{P})$ is the set of vertices of the polytope $\mathcal{P}$.

Therefore the relaxed problem is the following linear program

$$(LP) \begin{cases} \min \sum\limits_{i=1}^{n} \gamma_i y_i \\ s.t \\ \quad x \in \mathcal{P} \end{cases}$$

Remark 4.1. The only part of LP relaxation that depends on the received vector is the objective function.

4.2.2. *Decoding algorithm of LP decoder.* The decoding algorithm of the LP decoder using the proper polytope $\mathcal{P}$ is:

- Solving (LP).
  - If the solution is integral, output the corresponding codeword.
  - If the solution is fractional, output "error".

Notice that if LP decoder produce a codeword, it's guaranteed to be the codeword produced by the ML decoder, and we say that LP decoder has what we call *ML certificate property.* This property is one of the advantages of LP decoder.

4.2.3. *LP decoder properties.* The LP decoder will succeed, if and only if, the transmitted codeword is the unique optimal solution to the (LP). So we can write the probability of error, given a transmitted codeword y as:

$$Pr[error|y] = Pr[\exists f \in \mathcal{P},\ f \neq y\ :\ \sum_i \gamma_i f_i \leq \sum_i \gamma_i y_i]$$

In addition we have the following theorem

Theorem 4.2. [11] *Let $\mathbb{C}$ be a binary code, and $\mathcal{P}$ a proper polytope for $\mathbb{C}$. Then the LP decoder using $\mathcal{P}$ is successful, if at most $\lceil d_{frac}/2 \rceil - 1$ bits are flipped by binary-input-symmetric channel, where*

$$d_{frac} = \min\{\sum_{i=1}^{n} |y_i - f_i| \ s.t\ y \in \mathbb{C},\ f \in V(\mathcal{P}),\ and\ f \neq y\}.$$

The real d_{frac} is called the fractional distance of the polytope $\mathcal{P}$. Note that d_{frac} is a lower bound on the distance of $\mathbb{C}$.

Recall that a binary-input channel is symmetric if the output alphabet can be partitioned into pairs (a, a') such that

$$Pr[\tilde{y}_i = a | y_i = 0] = Pr[\tilde{y}_i = a' | y_i = 1],\ and$$

$$Pr[\tilde{y}_i = a|y_i = 1] = Pr[\tilde{y}_i = a'|y_i = 0].$$

Now we will define a property which allows us to analyze decoding algorithm. This property is the $\mathbb{C}$-symmetry.

A proper polytope $\mathcal{P}$ for a binary code $\mathbb{C}$ is $\mathbb{C}$-symmetric if for all f in $\mathcal{P}$ and codewords in $\mathbb{C}$, the point $f^{[y]}$ is also an element from $\mathcal{P}$, where $f_i^{[y]} = |f_i - y_i|$. We have the following theorem

Theorem 4.3. [11] *If a polytope $\mathcal{P}$ is proper for the binary code $\mathbb{C}$, and $\mathcal{P}$ is $\mathbb{C}$-symmetric, then $\mathbb{C}$ must be linear.*

We can see that the fractional distance is a tool measuring the capacity of decoding for the LP decoder. The $\mathbb{C}$-symmetry of a polytope allows to compute efficiently fractional distance. In fact if a polytope $\mathcal{P}$ is proper for a code $\mathbb{C}$, and is $\mathbb{C}$-symmetric, then $d_{frac} = \min\limits_{f\in V(\mathcal{P})} \sum\limits_{i=1}^{n} f_i$, which is computable in polynomial time.

Also, we can prove that as long as the polytope used in the LP decoder is $\mathbb{C}-$symmetric, we can make the all-zeros assumption, i.e., we assume that the codeword sent over the channel is the vector $y = 0^n$, and we have

$$Pr[error\ |y\ sent] = Pr[error\ |0\ sent].$$

4.2.4. *The non binary case.* This method of decoding is generalized to $2^m - ary$ codes (see [8] for intensive experiments). In fact we know that for every m, $\mathbb{F}_{2^m}$ is a vector space over $\mathbb{F}_2$, so we can obtain a binary code from a 2^m-ary code.

Let $\{\alpha_1,\ \alpha_2,\ ...,\ \alpha_m\}$ be a basis for $\mathbb{F}_{2^m}$ over $\mathbb{F}_2$. To every element a of $\mathbb{F}_{2^m}$ we can associate the string $a_1\ a_2\ ...\ a_m$, such that $a = \sum\limits_{i=1}^{m} \alpha_i a_i$. So if $y = y_1\ ...\ y_n$ a codeword from $\mathbb{C}$ a 2^m-ary $[n, k, d]$ code, and $y_i \longrightarrow y_{i1}\ y_{i2}\ ...\ y_{im}$ we can associate to y the mn binary vector

$$y \longrightarrow (y_{11}\ ...\ y_{1m})\ (y_{21}\ \ ...\ y_{2m})\ ...\ (y_{n1}\ \ ...\ y_{nm})$$

Let $\mathbb{C}^b$ be the binary code obtained from $\mathbb{C}$, and we note that if $\mathbb{C}$ is an $[n, k, d]$ code , then $\mathbb{C}^b$ is an $[nm, km, d']$ code with $d' \geq d$ (see [23], chap 8). Then we can prove:

Proposition 4.4. *Let $\gamma_{ij} = \ln \frac{Pr[\tilde{y}_{ij}|y_{ij}=0]}{Pr[\tilde{y}_{ij}|y_{ij}=1]}$. Decoding $\mathbb{C}$, a 2^m-ary code, by the ML decoder over a binary input memoryless channel, given $\tilde{y}$ an $nm-$string the received word, is equivalent to solving the problem:*

$$\begin{cases} \min \sum\limits_{i=1}^{n}\sum\limits_{j=1}^{m} \gamma_{ij} y_{ij} \\ s.t \\ \ y \in \mathbb{C}^b \end{cases}$$

By using $\mathbb{C}^b$ we can construct an LP relaxation for the ML decoder of $\mathbb{C}$. If the polytope generated is proper for $\mathbb{C}^b$, so we can say that the LP decoder have the ML certificate property. It's due to the fact that $\mathbb{F}_{2^m}$ is a $\mathbb{F}_2$-vector space. For the same reason we can generate the other properties (symmetry, ...).

4.2.5. *LP decoder for LDPC codes.* An LDPC code is a linear code that we have from a sparse bipartite graph. Assume a graph G, with n left nodes, called variable nodes, and m right nodes, called check nodes. This graph (which we call a Tanner [27] or factor graph) generate a linear code $\mathbb{C}$, of length n and dimension at least $n - m$. Every bit of a code word is associated with a variable node, and vector a $y = y_1\ y_2\ ...,\ y_n$ is an element of $\mathbb{C}$ if for all check nodes the sum of neighboring positions among the variable nodes, is zero modulo 2.

In [12] and [11] we find an LP relaxation for a binary LDPC code. The construction is a function of the factor graph representation of the graph, based on the parity polytope construction of [17].

Let G be a graph associated to an LDPC code $\mathbb{C}$, C the set check nodes and V the set of variable nodes. $N(j)$ denote the neighbors set of the node j. For all $j \in C$, we define $E_j = \{S \subseteq N(j)\ :\ |S|\ even\}$. We can notice that for every $S \in E_j$ if we set all bits in S to 1, and all the others in $N(j) \backslash S$ to 0, we obtain a vector that satisfies the check j.

For each variable node i, associate a variable f_i, and the cost γ_i, as it is defined before. For all $j \in C$ and $S \in E_j$, a variable w_{jS} is introduced, it serves to indicate if a word uses the configuration S to satisfy the check node j. Then the LP decoder for LDPC codes is:

$$\begin{cases} \min \sum\limits_i \gamma_i f_i & \\ s.t & \\ \quad \sum\limits_{S \in E_j} w_{jS} = 1 & \forall j \in C \\ \quad f_i = \sum\limits_{S \in E_j,\ S \ni i} w_{jS} & \forall\ edges\ (i,j) \\ \quad f_i \in [0,1] & \forall i \in V \\ \quad w_{iS} \in [0,1] & \forall j \in C\ and\ S \in E_j \end{cases}$$

Remark 4.5. The construction of the constraints can de done for all linear codes, but for a LDPC code the size of the polytope obtained is linear regarding the code length [11]. That is why this construction is associated to LDPC codes.

This polytope is proper and symmetric [11], so the LP decoder has the ML certificate property. In [11] there is a proof that for a factor graph with check degree at least 2, variable degree at least 3, and a girth at least 4, the fractional distance is at least $\Omega(n^{1-\epsilon})$ for girth $\Omega(\log n)$, where $\epsilon > 0$. This means that LP decoder can correct $\Omega(n^{1-\epsilon})$ errors.

4.3. Expander arguments

In this section we consider the binary case ($\mathbb{F} = \mathbb{F}_2$), and adversarial noise channel model in which case some of the bits are flipped arbitrarily. We say that a decoder corrects a fraction ε of error from code $\mathbb{C}$, when on input $w \in \mathbb{F}_2^n$, will outputs a codeword in the set $\{v : d(v, w) \leq \varepsilon n\}$. The class of expander codes is a subclass of LDPC codes based on (regular) expander graphs. Expander arguments were used in [6] to justify the excellent performance of decoding LDPC codes [15].

In [26], the authors introduced a class of asymptotically good linear codes. The code is based on a bipartite (c,d)-regular expander graph G and a subcode C_0 of block length d and minimum relative distance δ_0, which we denote by $\mathbb{C}(G, C_0)$. Then if G has expansion $(\alpha, c/d\delta_0)$ and rate $r = R(C_0) > (c-1)/c$, the code $\mathbb{C}(G, C_0)$ will have rate at least $cr-(c-1)$ and minimum relative distance at least α. Furthermore, a simple decoding algorithm called flipping decoding (at least in the case of C_0 being the parity code as in the preceding subsection) will correct an $\mathcal{O}(\alpha)$ fraction of error from $\mathbb{C}(G, C_0)$. We note also that the flipping decoding is in the complexity class NC^1 (see definition 1.6) and a LP decoder may be constructed [13]. These ideas are refined by the work of [31, 2] where the subcode C_0 is associated to the (neighbors) edges incident to each vertex of the bipartite graph, and also flipping algorithms are derived.

The above concepts suggests the following modification (extension) by introducing the chromatic number of an expander graph as follows. Suppose that $\chi(G) = k$. Then we may decode in rounds, where each round consists of k steps. In the i^{th} step, each vertex colored by i execute locally a ML decoding (for example). We note that each graph may be transformed to a bipartite graph (see [26, 18]). We also note that expander codes are nicely generalized to hypergraphs in [7] with bounds performance.

5. Questions

We ask some immediate questions:

1): A clear characterization of the class NC by CA. We note that this is related to the question : how many times the firing squad lemma is used?

2): Performance of the flipping decoding with the use of chromatic number, described in the last paragraph of subsection 4.3.

3): The use of the graphs (or hypergraphs) from [18] in the construction of codes on graphs with (a large chromatic number and) a large girth.

4): Experimental tests to confirm the performance of the LP decoder for non binary codes. This question is from [11], where also a LP decoder is proposed for dense Tanner graphs based on the polytope of [30].

5): The use of the new error correcting codes techniques in simulations between CA [21], and in the constructions of CA [22].

Acknowledgement

Thanks to J. Mazoyer for introducing us to one of his favorite subjects. Also our Acknowledgements to L. Lovász. We are indebted to B. Martin for stimulating discussions.

References

[1] N. Alon and J. H. Spencer. The probabilistic Method. *New York, Wiley, 1992.*

[2] A. Barg and G. Zémor. Error exponents of expander codes. *IEEE Trans. Inform. Theory, 48(6):1725-1729, 2002.*

[3] H. Ben-azza. Automates Cellulaires et pavages vus comme des Réseaux Booléens. *Thèse Lyon1, 1995.*

[4] A. Borodin. On Relating Time and Space to Size and Depth. *SIAM J. Comput, vol 6(4), 733-744,1977.*

[5] D. Burshtein and G. Miller. Expander graphs arguments for message-passing algorithms. *IEEE Trans. Inform. Theory, pages 782-790, February 2002.*

[6] E. R. Berlekamp, R. J. McEliece, and H. C. Van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory, vol 24 : 384–386, 1978.*

[7] Y. Bilu and S. Hoory. On codes from hypergraphs. *European J. Combin., 25(3):339-354, 2004.*

[8] M. C. Davey and D. J. C. MacKay. Low Density Parity Check Codes over GF(q). *IEEE Communications Letters, Vol 2, No. 6, June 1998.*

[9] M. Delorme. An introduction to Cellular Automata. *Research Report N^0 1998-37, ENS-Lip, Lyon, 1998.*

[10] P. Dunne. The complexity of Boolean Networks. *APIC, Number 29, Academic Press, 1988.*

[11] J. Feldman. Decoding Error-Correcting Codes via Linear Programming. *PhD thesis, MIT, 2003.*

[12] J. Feldman, R. Karger, and M.J Wainwright. LP Decoding. *In Proc. 41 st Annual Allerton Conference on Communication, Control, and Computating, 2003.*

[13] J. Feldman and C. Stein. LP Decoding Achieves Capacity. *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, pages 460-469, 2005.*

[14] P. Gács. Reliable computation with cellular automata. *Journal of Computer System Science 32, no. 1, 15-78, 1986.*

[15] R. Gallager. Low-density parity-check codes.*MIT press, Cambridge, MA, 1963.*

[16] M. Garey and D. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. *Freeman, New York, 1979.*

[17] R. G. Jeroslow. On defining sets of vertices of the hypercube by linear inequalities. *Discrete Mathematics, 11:119-124, 1975.*

[18] L. Lovász. On Chromatic Number of Finite Set-Systems. Acta Mathematica Academiae Scientiarum Hungaricae, Tomus 19 (1-2), pages 59-67, 1968.

[19] F.J. MacWilliams and N.J.A. Sloane. Theory of error-correcting codes. *North Holland, 1977.*

[20] J. Mazoyer. Problèmes des fusiliers. *Thèse, Lyon1, 1989.*

[21] C. Nichitiu and E. Rémila. Simulations of graph automata. *Research Report N^0 1998-44, ENS-Lip, Lyon, 1998.*

[22] N. Pippenger. Developments in " the synthesis of reliable organisms from unreliable components". *Proceedings of Symposia in Pure Mathematics, vol 50, 311-324, 1994.*

[23] S. Roman. Coding and Information Theory. *New York, Springer-Verlag, 1992.*

[24] W. L. Ruzzo. On Uniform Circuit Complexity. *Journal of Computer and System Sciences, 22 :365-383, 1981.*

[25] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal, vol. 27, pp. 379-423 and 623-656, July and October, 1948.*

[26] M. Sipser and D. Spielman. Expander codes. *IEEE Trans. Inform. Theory, vol 42 1710–1722, 1996*

[27] M. Tanner. A recursive approach to low complexity codes. *IEEE Trans. Inform. Theory, 27(5), 533-547, 1981.*

[28] A. Vardy. Algorithmic complexity in coding theory and the minimum distance problem. *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, pages 92-109, El Paso, Texas, 4-6 May 1997.*

[29] K. Wagner and G. Wechsung. Computational Complexity. *Reidel Publishing Company, 1985.*

[30] M. Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences, 43(3):441-466, 1991.*

[31] G. Zémor. On expander codes. *IEEE Trans. Inform. Theory, 47 (2):835-837, 2001.*

OPTIMAL TIME SELF-ASSEMBLY FOR SQUARES AND CUBES

CHRISTIANE BERCOFF

E-mail address: c.berco@univ-cezanne.fr

supported by LIF, 39 rue Joliot-Curie, 13013 Marseille, France

Abstract. We show that a finite union of rectilinear tiles pairwise disjoint, called $\mathbb{Z}$-set, is a tile in sense of Beauquier-Nivat. We consider $7q$-tiles which are pseudo-hexagons composed of q and six translated copies of it, where q is a given pseudo-hexagonal $\mathbb{Z}$-set. The contour word of such a $7^n q$-tile is defined by iterating a string rewriting system from the Beauquier-Nivat's factorization of the contour word of q into six words which codes the sides of q. This induces an efficient algorithm to construct geometrically the contour word of $7^n q$-tile. By definition, a $7^n q$-tile lets us make regular tilings of the plane by translation of a given pattern q.

Keywords: Rectilinear Tile, Replicating Fractile, Regular Tiling, Rewriting System

Journées Automates Cellulaires 2008 (Uzès), pp. 276-276

FROM UNDECIDABILITY TO RANDOMNESS: A NEW PARADIGM

SERGE GRIGORIEFF

LIAFA, CNRS & Université Paris 7
E-mail address: seg@liafa.jussieu.fr

We present variations of Chaitin's Ω number of the following form:

Let U be a machine which is "universal by prefix adjunction". The probability that its output be in some fixed set A is random (or random in some jump oracle) in the sense of Martin-Löf.

The notion of machine U "universal by prefix adjunction" we consider is as follows: U simulates on input $c(e)x$ the machine with code e on input x.

Our results deal with both finite and infinite computations.
Except for the case of finite computations and Σ_1^0 sets A, such complexity results about A refine uncomputability (or uncomputability in some jump oracle).

The difficulty of such results relies in the fact that Martin-Löf randomness is a fragile notion: modifications which would not modify undecidabilty may destroy randomness.

In the case of infinite computations, one of the main tools is the notion of effective Wadge reduction.

Some of our randomness results can be proved to fail with arbitrary universal machines.