



HAL
open science

On Factor Universality in Symbolic Spaces

Laurent Boyer, Guillaume Theyssier

► **To cite this version:**

Laurent Boyer, Guillaume Theyssier. On Factor Universality in Symbolic Spaces. 2010. hal-00474559v2

HAL Id: hal-00474559

<https://hal.science/hal-00474559v2>

Preprint submitted on 11 Jun 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On Factor Universality in Symbolic Spaces

Laurent Boyer^{*1} and Guillaume Theyssier^{**,** *1}

LAMA, (UMR 5127 — CNRS, Université de Savoie), Campus Scientifique,
73376 Le Bourget-du-lac cedex FRANCE

Abstract. The study of factoring relations between subshifts or cellular automata is central in symbolic dynamics. Besides, a notion of intrinsic universality for cellular automata based on an operation of rescaling is receiving more and more attention in the literature. In this paper, we propose to study the factoring relation up to rescalings, and ask for the existence of universal objects for that simulation relation.

In classical simulations of a system S by a system T , the simulation takes place on a specific subset of configurations of T depending on S (this is the case for intrinsic universality). Our setting, however, asks for every configurations of T to have a meaningful interpretation in S . Despite this strong requirement, we show that there exists a cellular automaton able to simulate any other in a large class containing arbitrarily complex ones. We also consider the case of subshifts and, using arguments from recursion theory, we give negative results about the existence of universal objects in some classes.

1 Introduction and definitions

Tilings and cellular automata are two paradigmatic models often considered in the fields of complex systems and natural computing. They are complementary —one is static and non-deterministic and the other is dynamic and deterministic— but they are both formally simple and both related to symbolic spaces. Moreover, many links are now established between the two models (see for instance [10,9]) so it is natural to consider them together.

Both are known to be Turing-powerful since their introduction in the mid-20th century [24,18]. However, analyzing their ability to process information only through translations into the Turing world is very restrictive. Such models of natural computing deserve a natural and intrinsic notion of reduction to compare their objects one to each other. Following this line of thought, several notions of simulations were proposed recently which are intrinsic to each model, and lead to corresponding intrinsic notions of universality [19,23,15,5]: a system is universal if it is able to simulate any other from the same class.

Intrinsic universality for cellular automata is probably the most studied of such notions [20,21,16,2,6]. The underlying relation of simulation uses uniform

* laurent.boyer@univ-savoie.fr

** guillaume.theyssier@univ-savoie.fr

*** Partially supported by French ANR 'projet blanc' EMC (NT09_555297)

encodings working at the level of blocks of cells. More precisely, S simulates T if S , when restricted to a suitable subset of 'correct' configurations, is isomorphic to T via such an encoding. Our approach is different and uses redundancy of information instead of restriction to a subset of configurations. In our setting, S simulates T if there is a uniform way of projecting *the whole* phase space of S onto the phase space of T (a precise definition is given below). The question addressed by this paper is the existence of universal objects with respect to that simulation relation, we call them *factor universal* objects.

The first contribution of this paper is the formalism based on the well-known mathematical notion of action: it allows to encompass both subshifts and cellular automata, it gives a new look at the notion of cell grouping which is the root of the simulation relation used in intrinsic universality, and it establishes connections with the work of Hochman [8] where the use of sub-actions is crucial. Our main result is that, although factor-universal objects do not generally exist (theorem 1), it can still be constructed for some large class like the set of cellular automata having a persistent state (theorem 2).

Basic definitions. Given a finite set Q and an integer $d \geq 1$, the *symbolic space* of *dimension* d over *alphabet* Q is the set $Q^{\mathbb{Z}^d}$. It can be seen as an infinite set of cells arranged as a lattice \mathbb{Z}^d and each carrying a value from Q . An element of $Q^{\mathbb{Z}^d}$ is called a *configuration*. $Q^{\mathbb{Z}^d}$ is naturally equipped with the compact Cantor topology [13] which is the product topology of the discrete topology on Q (it can also be defined via a metric).

Another key notion in the context of symbolic spaces is that of *finite patterns* that may occur in infinite configurations. For our purpose, rectangular patterns will be enough. Given $\mathbf{z} = (z_1, \dots, z_d) \in \mathbb{Z}^d$ with $z_i > 0$ for all i , the hyperrectangle $\mathcal{R}_{\mathbf{z}}$ is the set of vectors $\mathbf{z}' = (z'_1, \dots, z'_d) \in \mathbb{Z}^d$ such that $0 \leq z'_i < z_i$ for all i . A Q -pattern of shape $\mathcal{R}_{\mathbf{z}}$ is a coloring of $\mathcal{R}_{\mathbf{z}}$ by Q , that is an element of $Q^{\mathcal{R}_{\mathbf{z}}}$. Given a configuration $c \in Q^{\mathbb{Z}^d}$, the pattern of shape $\mathcal{R}_{\mathbf{z}_s}$ extracted from c at position $\mathbf{z}_p \in \mathbb{Z}^d$, denoted by $\mathcal{P}_{\mathbf{z}_p}^{\mathbf{z}_s}(c)$, is simply: $z \in \mathcal{R}_{\mathbf{z}_s} \mapsto c(\mathbf{z}_p + z)$.

The objects we study (subshifts and cellular automata) share the property of being uniform, *i.e.* invariant by translations. Formally, given $\mathbf{z} \in \mathbb{Z}^d$ the translation of vector \mathbf{z} , denoted $\sigma_{\mathbf{z}}$, is the function mapping a configuration $c \in Q^{\mathbb{Z}^d}$ to the configuration $\sigma_{\mathbf{z}}(c)$ such that $\forall \mathbf{z}' \in \mathbb{Z}^d, \sigma_{\mathbf{z}}(c)(\mathbf{z}') = c(\mathbf{z}' + \mathbf{z})$.

A *subshift* is a subset of $Q^{\mathbb{Z}^d}$ which is translation invariant and closed for the Cantor topology. Equivalently, a subshift is a set Σ_L of configurations avoiding any occurrence of any finite pattern from a given language of patterns L :

$$\Sigma_L = \{c \in Q^{\mathbb{Z}^d} : \forall \mathbf{z}, \mathbf{z}' \in \mathbb{Z}^d \text{ with } z_i > 0 \text{ for all } i, \mathcal{P}_{\mathbf{z}'}^{\mathbf{z}}(c) \notin L\}.$$

A subshift of *finite type* is a subshift of the form Σ_L where L is finite. There are strong connections between subshifts of finite type in dimension 2 and sets of tilings generated by a set of wang tiles. In particular, due to Berger's theorem [1], it is undecidable, given a finite L , to determine whether Σ_L is empty or not.

A *cellular automaton* is a local and uniform map on a symbolic space. Formally, it is given as a 4-tuple by its dimension d , its alphabet Q , its *neighborhood*

$V \subseteq \mathbb{Z}^d$ (finite) and its *local transition map* $f : Q^V \rightarrow Q$. To that formal object we associate a *global map* F acting on $Q^{\mathbb{Z}^d}$ as follows:

$$\forall c \in \mathbb{Z}^d, \forall z \in \mathbb{Z}^d, F(c)(z) = f(z' \in V \mapsto c(z + z')).$$

The fundamental theorem of Curtis-Lyndon-Hedlund [7] states that global maps of cellular automata are exactly continuous maps on symbolic spaces which commute with translations.

Actions and rescalings. Let $(\mathbb{M}, +)$ be a monoid (a set equipped with an associative law and a neutral element). An \mathbb{M} -*action* on a space X is a function $\Psi : \mathbb{M} \times X \rightarrow X$ such that $\Psi(0, x) = x$ (for all $x \in X$ and 0 being the neutral element of \mathbb{M}) and

$$\forall x \in X, \forall m, m' \in \mathbb{M}, \Psi(m + m', x) = \Psi(m, \Psi(m', x)).$$

We will use the formalism of action to study both subshifts and cellular automata:

- if $\Sigma \subseteq Q^{\mathbb{Z}^d}$ is a subshift, we canonically associate to it the \mathbb{Z}^d -action Ψ_Σ on Σ defined by $\Psi_\Sigma(z, x) = \sigma_z(x)$;
- if F is a cellular automaton on the space $Q^{\mathbb{Z}^d}$, we canonically associate to it the $\mathbb{N} \times \mathbb{Z}^d$ -action Ψ_F on $Q^{\mathbb{Z}^d}$ defined by $\Psi_F((t, z), x) = \sigma_z \circ F^t(x)$.

If \mathbb{M}' is a sub-monoid of \mathbb{M} , Ψ induces an \mathbb{M}' -action by restriction to the domain $\mathbb{M}' \times X$. \mathbb{M} and \mathbb{M}' can be isomorphic or not and both cases might be interesting. For instance, studying a cellular automaton F as a classical dynamical system consists in forgetting the spacial component of Ψ_F and focusing on the pure temporal action of F . This point of view was often adopted in the literature (*e.g.*, topological dynamics of cellular automata [13]) but, interestingly enough, recent work of Sablik [22] tends to re-incorporate the spacial component of actions to better study the dynamics of cellular automata.

In this paper, we will only consider the case where \mathbb{M} and \mathbb{M}' are isomorphic. More precisely, in our context, \mathbb{M} will be of the form \mathbb{Z}^d or $\mathbb{N} \times \mathbb{Z}^d$ and we will consider sub-monoids of the form $\mathbb{M}' = t_0\mathbb{N} \times z_1\mathbb{Z} \times \dots \times z_d\mathbb{Z}$, with $t_0 > 0$ and $z_i > 0$ for all i . In this case, passing from the \mathbb{M} -action to the \mathbb{M}' -action can be seen as a neutral change of point of view on the system that we call *rescaling* in the sequel. The intuition is that we change the discrete units of time and space, passing from 1 to t_0 in time and 1 to z_i in direction i . Given a subshift or a cellular automaton, a *scaled action* is simply the restriction of their canonical action to some sub-monoid of the form \mathbb{M}' . It is worth noticing that a scaled action associated to a subshift (resp. a cellular automaton) on the alphabet Q is always isomorphic to the canonical action of a subshift (resp. a cellular automaton) on an alphabet of the form Q^k . More concretely, this isomorphism comes from the natural one-to-one map from $Q^{\mathbb{Z}^d}$ to $(Q^{\mathcal{R}_{z_s}})^{\mathbb{Z}^d}$, where $z_s = (z_1, \dots, z_d)$, which maps a configuration c to: $z \mapsto \mathcal{P}_{z \times z_s}^{z_s}(c)$, where the operation \times on \mathbb{Z}^d denotes coordinate-wise multiplication. Our notion of rescaling for cellular automata is similar to the one in [19,23] which is the basic ingredient to define intrinsic universality.

Factors. One of the central notion in symbolic dynamics is that of *factor*. Intuitively, a factor is a uniform continuous projection. This notion has also been used with success in the study of expansive cellular automata [17] and more generally as a classification tools for cellular automata [12,3]. As we study both multi-dimensional subshifts and cellular automata, we give a unified definition using the formalism of actions.

Definition 1 Let \mathbb{M} and \mathbb{M}' be isomorphic monoids via $i : \mathbb{M} \rightarrow \mathbb{M}'$. We say an \mathbb{M}' -action ϕ' on X' is a factor of a \mathbb{M} -action ϕ on X if there is a continuous onto map $\pi : X \rightarrow X'$ such that: $\forall x \in X, \forall m \in \mathbb{M}, \pi(\phi(m, x)) = \phi'(i(m), \pi(x))$.

Two key points are that: (1) any orbit in (ϕ, X) projects onto some orbit of (ϕ', X') via π , and (2) any orbit of ϕ' can be realized as such a projection. In a word, the simulation of (ϕ', X') by (ϕ, X) is *everywhere meaningful* and *complete*.

2 Factor Universality

At this point, we could compare subshifts or cellular automata through the factoring relation between their canonical actions, saying that system S factors onto system T if the canonical action of S factors onto that of T . However, this gives an excessive importance to the alphabet and forbid the existence of universal objects due to entropy considerations (factoring cannot increase entropy). In [8], this limitation is bypassed via dimension changes: a d -dimensional system is compared to k -dimensional systems ($k < d$) via its k -dimensional sub-actions. Our point of view is different. We always work at constant dimension, but we use another kind of sub-actions: *scaled actions* defined above. For a fixed dimension, monoids of scaled actions are all isomorphic and we will consider only canonical component-wise isomorphisms between them. We can now formulate the central definition of the paper.

Definition 2 Let S and T be two d -dimensional subshifts (resp. CA). We say that T is simulated by S , denoted $T \preceq S$, if some scaled action of S factors onto some scaled action of T .

As usual when working on symbolic spaces, continuity and uniformity implies locality (Curtis-Lyndon-Hedlund theorem [7]). In our context of rescalings, the locality is no longer expressed at the level of cells, but at the level of groups of cells. More precisely, we say that a map $\phi : Q_1^{\mathbb{Z}^d} \rightarrow Q_2^{\mathbb{Z}^d}$ is *local* if there exist: $r \in \mathbb{N}$ (locality radius), two shapes \mathcal{R}_{z_1} and \mathcal{R}_{z_2} (source and destination scales), and a local function $f : Q_1^{\mathcal{R}_{(2r+1)z_1}} \rightarrow Q_2^{\mathcal{R}_{z_2}}$ such that

$$\forall c \in Q_1^{\mathbb{Z}^d}, \forall z \in \mathbb{Z}^d, \mathcal{P}_{z \times z_2}^{z_2}(\phi(c)) = f(\mathcal{P}_{z \times z_1 - rz_2}^{(2r+1)z_1}(c)).$$

To fix ideas, if $d = z_1 = z_2 = 1$ and $Q_1 = Q_2$, f is just the local map of a cellular automaton of radius r and ϕ is its corresponding global map.

Proposition 1 Fix a dimension d . Let Σ_1 and Σ_2 be two d -dimensional subshifts and let F_1 and F_2 be two d -dimensional CA of alphabet Q_1 and Q_2 respectively. Then we have:

- $\Sigma_2 \preceq \Sigma_1$ if and only if there is a local map ϕ such that $\phi(\Sigma_1) = \Sigma_2$;
- $F_2 \preceq F_1$ if and only if there is an onto local map ϕ from $Q_1^{\mathbb{Z}^d}$ to $Q_2^{\mathbb{Z}^d}$ and integers $t_1, t_2 \in \mathbb{N}$ such that $\phi \circ F_1^{t_1} = F_2^{t_2} \circ \phi$.

Besides the work of Hochman [8], notions of simulations similar to \preceq have already been considered for tilings [15] or for cellular automata [23,4] or more general settings [14]. Each time, one of the main concern is the existence of universal objects: this is precisely the central point of the present paper.

Definition 3 Let \mathcal{C} be a class of subshifts (resp. cellular automata). A subshift (resp. cellular automaton) U is \mathcal{C} -universal if $U \in \mathcal{C}$ and $X \preceq U$ for any $X \in \mathcal{C}$.

Whatever the fixed dimension, there is no universal subshift for cardinality reasons: there are uncountably many subshifts but for a given subshift U there are at most countably many different subshifts \preceq -simulated by U (by proposition 1). The following theorem uses recursion theoretic arguments to yield other negative results concerning universality (similar arguments were used in [19,8] in different settings).

Theorem 1 Fix a dimension $d \geq 2$. Then there is no universal subshift of finite type of dimension d and there is no surjective-universal CA of dimension d .

3 A Large Class with a Universal Object

In this section, we restrict to dimension 1 to make a clear exposition of the main result (theorem 2).

Definition 4 A CA \mathcal{A} is said to be persistent if there is a state $q_0 \in Q_{\mathcal{A}}$ such that for any configuration $c \in Q_{\mathcal{A}}^{\mathbb{Z}}$ if $c(i) = q_0$ then $\mathcal{A}(c)(i) = q_0$.

We denote by \mathbb{P} the set of all persistent CA.

Note that for any CA, you may add an extra persistent state and obtain a CA in \mathbb{P} containing the dynamics of the first one.

Theorem 2 There exists a \mathbb{P} -universal cellular automaton.

In the following, we describe such a \mathbb{P} -universal CA \mathcal{U} with radius 1 and alphabet $Q_{\mathcal{U}}$.

We denote by \mathbb{P}_0 the set of CA of \mathbb{P} with radius 1 and alphabet of size 2^p for some $p \in \mathbb{N}$. One may easily describe for any CA $\mathcal{B} \in \mathbb{P}$ a CA $\mathcal{A} \in \mathbb{P}_0$ such that $\mathcal{B} \preceq \mathcal{A}$. Using transitivity of \preceq , it will be sufficient, in order to prove \mathbb{P} -universality of \mathcal{U} , to exhibit for any CA $\mathcal{A} \in \mathbb{P}_0$ an onto local map $\phi_{\mathcal{A}}$ from $Q_{\mathcal{U}}^{\mathbb{Z}}$ to $Q_{\mathcal{A}}^{\mathbb{Z}}$ and an integer $\tau_{\mathcal{A}}$ such that $\mathcal{U}^{\tau_{\mathcal{A}}} \circ \phi_{\mathcal{A}} = \phi_{\mathcal{A}} \circ \mathcal{A}$.

To do so, for each \mathcal{A} , we introduce an integer $l_{\mathcal{A}}$ and a dichotomy on words of $Q_{\mathcal{U}}^{l_{\mathcal{A}}}$.

- on the one side we have what we call \mathcal{A} -correct macrocells (or \mathcal{A} -macrocells). They encode information about a current state $x \in Q_{\mathcal{A}}$, about the local rule of \mathcal{A} , and a machinery used to apply this rule to update the current state. In almost any case, they will be interpreted through $\phi_{\mathcal{A}}$ as x .
- on the other side we call all the other patterns \mathcal{A} -incorrect, and they will be interpreted as the persistent state of \mathcal{A} .

The idea behind the local rule of \mathcal{U} is to make every \mathcal{A} -macrocell determine if it is surrounded by other \mathcal{A} -macrocells. If this is the case, then interaction is possible, and the current state of the neighbor will be taken into account to compute the new current state, following the rule of \mathcal{A} . Otherwise, there is no interaction, the \mathcal{A} -macrocell evolves considering every \mathcal{A} -incorrect neighborhood as a persistent state neighbor. The difficulty is that although correctness is related to the particular CA being simulated, every configuration must evolve correctly for every possible CA.

The proof of universality uses the combination of two key properties: on one hand, correct patterns remain correct and evolve according to the rule being simulated, even if not surrounded by correct patterns (lemma 4); on the other hand, incorrect patterns are interpreted as the persistent state and never become correct (lemma 5).

To make the construction of \mathcal{U} readable, we describe its state set as a superposition of several layers: the main layer M contains most of the information about the simulation and the macrocells informations; *signals* layers are used to manage the evolution of the main layer; and *clock* layers guarantee synchronizations.

Correct macrocells description In the following we consider a simulated CA $\mathcal{A} \in \mathbb{P}_0$ with radius 1 and state set $Q_{\mathcal{A}}$ of size $n = 2^p$. We use a canonical binary enumeration of the state set, in which the first word ($0^{\log(n)}$) represents a persistent state of \mathcal{A} , denoted $p_{\mathcal{A}}$. Our \mathcal{A} -correct macrocells will be words of length $l_{\mathcal{A}}$ (specified above) whose main layer follow the pattern

$$\# C_i |Transition\ table| |State| |memory| \#$$

- $\#$ are delimiters, they never appear or disappear during the computation
- C_i is the *control state* used to control the successive steps of computation.
- $|Transition\ table|$ is the binary description of the transition table of \mathcal{A} .
- $|State|$ contains the binary value of the current state of the macrocell.
- $|memory|$ is a binary area which will be used to keep the values of the neighbors' current states before computing the new current state of the macrocell.

$|Transition\ table|$, $|State|$ and $|memory|$ are encoded with disjoint binary sub-alphabets. A cell whose state belongs to one of those sub-alphabets will never change sub-alphabet. Moreover, the states of the transition table's cells are never modified.

The current state description is $\log(n)$ -bits-long. In the transition table, images are ordered canonically, so the length of the description is simply $n^3 \log(n)$.

The memory should be at least $2\log(n)$ -bits-long in order to contain current state values of the two neighbors. But in order to simplify some proofs, we chose $l_{\mathcal{A}}$ such that it is at least half the total size of the macrocell, and such that the function $\mathcal{A} \rightarrow l_{\mathcal{A}}$ is one-to-one.

Most of the computation will happen on those very constrained patterns. In the next definition, we add an extra constraint on the control state to obtain \mathcal{A} -correct macrocells.

By stability of the sub-alphabets, such a correct \mathcal{A} -macrocell remains correct, but the state value may change. This is why we take some care when defining the current state value associated to the macrocell.

Definition 5 *A word $u \in Q_{\mathcal{U}}^{l_{\mathcal{A}}}$ of length $l_{\mathcal{A}}$ is said to be a \mathcal{A} -correct macrocell, denoted by $u \in \mathcal{C}_{\mathcal{A}}$, if its main layer follows the structure defined above (correct sub-alphabets for each cell, and correct transition table of \mathcal{A}), and if its control state is in C_0 .*

For each such \mathcal{A} -correct macrocell u , we define its associated state value $v(u) \in Q_{\mathcal{A}}$, which is the state described by its current state value after $l_{\mathcal{A}}$ steps of computation by \mathcal{U} . And this state value $v(u)$ only depends on u .

To ensure that the state value only depends on u , the memory area is used as a buffer to prevent modification of the current state value coming from the left, before the computation has been initialized. Details will be given in the following.

By extension we may sometime call \mathcal{A} -macrocells words following the general pattern, even with non- C_0 control state, in particular when they are images of a \mathcal{A} -correct macrocell.

The local rule. We describe the local behavior of \mathcal{U} starting from a correct \mathcal{A} -macrocell. The local rule will first determine which neighbors it may interact with (*Check of the neighbor's length and synchronization, and Transition table and state encoding check*), and then compute its new current state according to the rule of \mathcal{A} and eventually the value of those neighbors (*New current state computation*).

In order to guarantee the synchronization between \mathcal{A} -macrocells, we specify the duration of each step, and even of some sub-steps. It is done by a clock, which use specific layers of the states; their existence is proved by the following lemma:

Lemma 1 *For any $k, h \in \mathbb{N} \setminus \{1\}$, there exist a CA, and two states q_s , and q_f such that the leftmost cell of an area delimited by two $\#$ separated by $l - 2$ cells turns to state q_f at some time $t > k \cdot l^2 + h \cdot l$ iff this cell was in state q_s exactly $k \cdot l^2 + h \cdot l$ steps before. Moreover, this property is guaranteed independently of what is outside the two $\#$.*

At the beginning of each step, the control state C_i will turn to C_{i+1} , initiate the corresponding clock, and initiate some signals which will manage the

evolution. Those signals are distinct states propagating on upper layers of the configuration, and interacting with the main layer and other signals. We say that a signal *belongs* to a macrocell if it was generated in this macrocell's area, between the two $\#$. And, thanks to our evolution rule, a signal always knows if it is in its cell or in the area to the right or left of its cell. It is also useful sometimes to make signals carry one extra bit of information. It is simple to do it using distinct states, since the number of bits is bounded.

Check of the neighbor's length and synchronization. $C_0 \rightarrow C_1$ (that is to say that when the control cell's state is C_0 it becomes C_1):

Recall that we are interested to the behavior in the case of an \mathcal{A} -correct macrocell. When C_0 becomes C_1 , it initializes two *control bits* with value 0, in the main layer of the control cell, and it launches signals. Since the construction is classical, we simply illustrate the desired behaviors by figure 1. Those two pictures illustrate the signal machinery in the case of respectively left and right neighbors of same length and with state C_0 appearing simultaneously (what we call synchronized). Every transition whose image is one of the signal involved in this checking appears on those pictures.

The first signals s_1 and s_4 erase all signals belonging to our macrocell. Together with the length of the memory being bigger than the length between the control state where signals are generated and the end of the current state area, it constitutes the protection of the current state value from erratic signals coming from outside the macrocell, and justifies that in definition 5 $v(u)$ is a function of u .

If the neighbors have same length and are synchronized, this whole step takes 4 times the length of the macrocell, $l_{\mathcal{A}}$. After $4 \cdot l_{\mathcal{A}}$ steps, the control state C_1 becomes C_2 , and if it did not receive a positive result from one side, it concludes that the involved neighbor is incorrect. This is managed using a clock signal (with $h = 2$ and $k = 0$ in lemma 1) initialized by C_0 on a specific layer. When q_f is raised on this layer, C_1 becomes C_2 . The important point is that we ensure the following property.

Lemma 2 *The control state of an \mathcal{A} -correct macrocell becomes C_2 exactly $4 \cdot l_{\mathcal{A}}$ steps after C_0 appeared. At this step each control bit has turned to 1 iff the corresponding neighboring macrocell has same length and synchronisation than the considered macrocell.*

The proof of this lemma is direct for the length but asks to enter into some more (simple but fastidious) details for the synchronization part.

Transition table and state encoding check. $C_2 \rightarrow C_3$:

In this step, for each neighboring pattern with same length and synchronization, the macrocell checks whether the transition table and the current state are compatible with its own (same lengths, and same content for the transition table) or not.

When C_2 appears it launches the following test for each neighbor whose corresponding control bit was 1, and initializes two fresh bits to 0. First, a signal

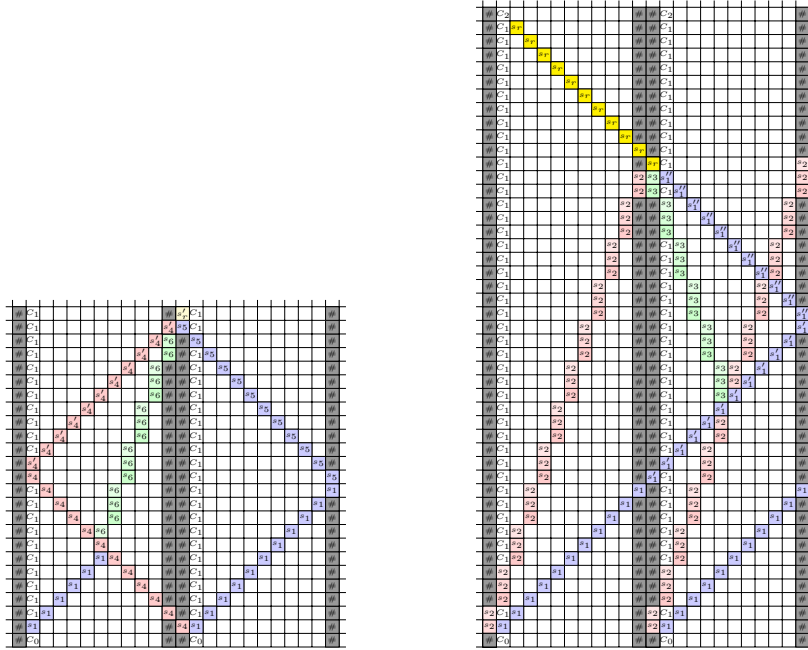


Fig. 1. Successful left (resp. right) neighbor test by the right (resp. left) macrocell (mix of main and signal layers for easier reading)

is generated and puts a *mark* (that is to say a non-moving signal) on the first cell of the transition table of its macrocell, and another mark on the first cell of the transition table of the neighbor it checks. Then signals are exchanged between those two marks that will each time carry the binary state of the cell pointed by one mark to the next unchecked cell of the other macrocell; it compares this cell's binary state to the carried binary state, and push the mark by two cells. If no difference is detected and if both marks reach the end of the transition tables simultaneously, a correctness signal is sent to the control state.

After the transition table has been checked, the same mechanism is used to check that the current state encoding areas have same length. At the end of those tests, the results are sent to the control cell which again keeps the information on two control bits. For each cell of the transition table or the current state, checking takes $2 \cdot l_A$ steps. So checking a whole neighbor takes less than $2 \cdot l_A^2$. Again, a clock is used to make this test last exactly $2 \cdot l_A^2$ steps. Then the control cell is turned to C_4 .

Lemma 3 *The control state of our macrocell becomes C_4 exactly $2 \cdot l_A^2$ steps after C_2 appeared. At this step each control bit has turned to 1 iff the corresponding neighboring pattern has same length, synchronization, and if the lengths of the transition tables and current states, and the content of the transition table are equal. In this case we say that this pattern is compatible with our \mathcal{A} -macrocell.*

The proof of this lemma is straightforward. Keep in mind that some signals erased all erratic signals that could interact with our cell at a previous step.

New current state computation. $C_4 \rightarrow C_5$:

After all the tests have been done, the new current state has to be computed. We need to explicit how we consider the neighboring pattern. In the following, what we call *detected state* of one such pattern by our macrocell will be: either the persistent state if the neighbor is non-compatible with our \mathcal{A} -macrocell, or its current state if this is a compatible \mathcal{A} -macrocell.

At first, the detected states of the left and right neighbors are written to the memory. It is written in the binary memory alphabet. Each detected state is written on $\log(n)$ cells. If one neighbor is compatible, we copy its current state value to the memory using marks and signals similarly to the previous step. If it is not compatible, we write $0^{\log(n)}$, the length being the same as that of the current state area. We add a clock to specify that copies last exactly $2 \cdot l_{\mathcal{A}}^2$ steps, the neighbor being correct or not.

After $2 \cdot l_{\mathcal{A}}^2$ additional steps, the search for the image in the transition table starts. It consists in reading the binary word formed by the three image states (the current state of the cell followed by the two detected states copied in the memory), and turning it into a unary position in the transition table. It is then possible to place a mark at this position, and finally copy this pointed state to the current state area. We make the reading of the position last $4 \cdot l_{\mathcal{A}}^2$ steps. And copying the new state lasts $2 \cdot l_{\mathcal{A}}^2$. After the whole computation step, which lasts $7 \cdot l_{\mathcal{A}}^2$, the control state turns to C_5 .

Finally one step of simulation is completed after exactly $\tau_{\mathcal{A}} = 9 \cdot l_{\mathcal{A}}^2 + 4 \cdot l_{\mathcal{A}}$ steps. After this time the control state turns to C_5 .

To become C_0 again, and launch a new step of computation, we add another condition. We ask a clock launched exactly $\tau_{\mathcal{A}}$ steps before to raise a flag. And obviously this clock may only be launched by C_0 . It is realized using again signals of the lemma 1 computing on one more layer.

The state set of the universal CA is given by $Q_{\mathcal{U}} = M \times S \times C \cup \{C_f\}$ with

- the main layer : $M = \{C_0, C_5\} \cup \{C_i\}_{i \in \{1, \dots, 4\}} \times \{0, 1\}^2 \cup \{0_i, 1_i\}_{i \in \{tt, cs, m\}}$
- the signals layers : $S = \times_{i \in I} \{s_i\} \times \times_{j \in J} (\{s_j\} \times \{0, 1\})$
- the clocks layers (see lemma 1), one for each duration needed. $C = (\{0, 1\} \times \{s_i\}_{i \in I_c})^4$
- C_f is a single persistent state ensuring that $\mathcal{U} \in \mathbb{P}$

Yet, the transition rule of \mathcal{U} is partially specified, we call correct transitions those defined up to now, in the case of correct macrocells. But the other transitions may not be chosen arbitrarily. We specify the following behaviors:

- C_f is never modified by any transition
- the main layer is never modified by a non-correct transition: they act as the identity on the main layer.
- concerning the signal layer, apart from the collisions corresponding to the behavior described in the previous steps, all signals may cross each other

(each kind of signal is evolving on its own layer). However, except for transitions involved in the behavior described above, any signal that crosses a # is destroyed.

Interpretation We now describe the continuous onto map $\phi_{\mathcal{A}} : Q_{\mathcal{U}}^{\mathbb{Z}} \rightarrow Q_{\mathcal{A}}^{\mathbb{Z}}$ associated to \mathcal{A} . This map is induced by a local map $f_{\mathcal{A}}$ from patterns of shape $l_{\mathcal{A}}$ to individual states of \mathcal{A} . More precisely, using notation from proposition 1, we have $r = 0$, $z_1 = l_{\mathcal{A}}$, $z_2 = 1$, $t_2 = 1$ and $t_1 = \tau_{\mathcal{A}}$.

If $p_{\mathcal{A}}$ is the persistent state of \mathcal{A} , the local map $\psi_{\mathcal{A}}$ is defined as follows:

1. $\forall u \notin \mathcal{C}_{\mathcal{A}}, f_{\mathcal{A}}(u) = p_{\mathcal{A}}$
2. $\forall u \in \mathcal{C}_{\mathcal{A}}, f_{\mathcal{A}}(u) = v(u)$, with $v(u)$ the value from definition 5

Proof of theorem 2 The proof of the theorem relies on the two following lemmas. They are consequences of the construction, the intermediate lemmas and the clock lemma.

Lemma 4 $\forall c \in Q_{\mathcal{U}}^{\mathbb{Z}}, t_0 \in \mathbb{N}$, if $U^{t_0}(c)_{[0, l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$, then $v = U^{t_0 + \tau_{\mathcal{A}}}(c)_{[0, l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$, and $\psi_{\mathcal{A}}(v) = \delta_{\mathcal{A}}(\psi_{\mathcal{A}}(c_{[-l_{\mathcal{A}}, -1]}), \psi_{\mathcal{A}}(c_{[0, l_{\mathcal{A}}-1]}), \psi_{\mathcal{A}}(c_{[l_{\mathcal{A}}, 2 \cdot l_{\mathcal{A}}-1]}))$.

Lemma 5 If $\exists t \geq \tau_{\mathcal{A}}, c \in Q_{\mathcal{U}}^{\mathbb{Z}}$ such that $u = U^t(c)_{[0, l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$ then $v = U^{t - \tau_{\mathcal{A}}}(c)_{[0, l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$.

We can finally prove our main claim: $\forall \mathcal{A} \in \mathbb{P}_0, \mathcal{A} \preceq \mathcal{U}$. We use the characterization of proposition 1. Let $\mathcal{A} \in \mathbb{P}_0$ the associated length $l_{\mathcal{A}}$ and function $\phi_{\mathcal{A}}$ are defined as explained before. First, $\phi_{\mathcal{A}}$ is local (by definition) and onto, because correct macrocells are enough to encode any state of \mathcal{A} and thus concatenations of correct macrocells allow to encode any configuration of \mathcal{A} . Second, we have $\phi_{\mathcal{A}} \circ U^{\tau_{\mathcal{A}}} = \mathcal{A} \circ \phi_{\mathcal{A}}$. To see this we discuss on the pattern of shape $\mathcal{R}_{l_{\mathcal{A}}}$ at position 0 and the rest follows by translation. If this pattern is not in $\mathcal{C}_{\mathcal{A}}$ its image after $\tau_{\mathcal{A}}$ steps remains out of $\mathcal{C}_{\mathcal{A}}$ (lemma 5). If conversely this central word belongs to $\mathcal{C}_{\mathcal{A}}$, lemma 4 gives the desired property.

4 Perspectives

A natural extension of our work could be to generalize the construction to cellular automata having an equicontinuous point. The idea would be to use blocking words as a replacement for the persistent state. But it seems much harder, if not impossible.

Besides, the main open question left by this paper is the existence of universal CA. We conjecture that they do not exist and more precisely that no CA can simulate all products of shifts. A possible way to obtain this negative result would be to study limit sets: by a compactness argument, one can show that a universal CA must have a universal limit set. The main obstacle is that subshifts that are limit sets of CA are not well characterized.

Finally, we also leave open the existence of universal SFT and universal surjective CA in dimension 1.

References

1. R. Berger. The undecidability of the domino problem. *Mem. Amer. Math. Soc.*, 66, 1966.
2. Laurent Boyer and Guillaume Theyssier. On local symmetries and universality in cellular automata. In *STACS*, pages 195–206, 2009.
3. Julien Cervelle, Enrico Formenti, and Pierre Guillon. Ultimate traces of cellular automata. In *STACS*, pages 155–166, 2010.
4. Marianne Delorme, Jacques Mazoyer, Nicolas Ollinger, and Guillaume Theyssier. Bulking ii: Classifications of cellular automata. *CoRR*, abs/1001.5471, 2010.
5. David Doty, Jack H. Lutz, Matthew J. Patitz, Scott M. Summers, and Damien Woods. Intrinsic universality in self-assembly. In *STACS*, pages 275–286, 2010.
6. Jérôme Olivier Durand-Lose. Intrinsic universality of a 1-dimensional reversible cellular automaton. In *STACS*, pages 439–450, 1997.
7. G. A. Hedlund. Endomorphisms and Automorphisms of the Shift Dynamical Systems. *Mathematical Systems Theory*, 3(4):320–375, 1969.
8. Michael Hochman. A note on universality in multidimensional symbolic dynamics. *Discrete Contin. Dyn. Syst. Ser. S*, 2(2):301–314, 2009.
9. Michael Hochman. On the dynamics and recursive properties of multidimensional symbolic systems. *Inventiones Mathematicae*, 176(1):131–167, 2009.
10. J. Kari. The Nilpotency Problem of One-dimensional Cellular Automata. *SIAM Journal on Computing*, 21:571–586, 1992.
11. J. Kari. Reversibility and Surjectivity Problems of Cellular Automata. *Journal of Computer and System Sciences*, 48(1):149–182, 1994.
12. P. Kůrka. Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory and Dynamical Systems*, 17:417–433, 1997.
13. P. Kůrka. *Topological and symbolic dynamics*. Société Mathématique de France, 2003.
14. Petr Kurka. Zero-dimensional dynamical systems, formal languages, and universality. *Theory Comput. Syst.*, 32(4):423–433, 1999.
15. Grégory Lafitte and Michael Weiss. An almost totally universal tile set. In *TAMC*, pages 271–280, 2009.
16. Andrés Moreira. Universality and decidability of number-conserving cellular automata. *Theor. Comput. Sci.*, 292(3):711–721, 2003.
17. M. Nasu. The dynamics of expansive invertible onesided cellular automata. *Trans. Amer. Math. Soc.*, 354:4067–4084, 2002.
18. J. Von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, Illinois, 1966.
19. N. Ollinger. *Automates Cellulaires : structures*. PhD thesis, École Normale Supérieure de Lyon, décembre 2002.
20. N. Ollinger. The quest for small universal cellular automata. In *ICALP*, pages 318–330. Lecture Notes in Computer Science, 2002.
21. N. Ollinger. The intrinsic universality problem of one-dimensional cellular automata. In *STACS*, pages 632–641. Lecture Notes in Computer Science, 2003.
22. Mathieu Sablik. Directional dynamics for cellular automata: A sensitivity to initial condition approach. *Theor. Comput. Sci.*, 400(1-3):1–18, 2008.
23. G. Theyssier. *Automates Cellulaires : un modèle de complexités*. PhD thesis, École Normale Supérieure de Lyon, décembre 2005.
24. H. Wang. Proving theorems by pattern recognition ii. *Bell System Tech. Journal*, 40(2), 1961.

A Proofs from section 2

Proof (Proposition 1). First, an onto local map from Σ_1 to Σ_2 with shapes \mathcal{R}_{z_1} and \mathcal{R}_{z_2} induces a factoring relation from the \mathbb{M} -scaled action of Σ_1 onto the \mathbb{M}' -scaled action of Σ_2 with

$$\mathbb{M} = (z_1)_1\mathbb{Z} \times \cdots (z_1)_d\mathbb{Z}$$

and

$$\mathbb{M}' = (z_2)_1\mathbb{Z} \times \cdots (z_2)_d\mathbb{Z}.$$

Conversely, suppose that the relation $\Sigma_1 \preceq \Sigma_2$ is realized by a factor map π from the \mathbb{M} -scaled action of Σ_1 onto the \mathbb{M}' -scaled action of Σ_2 with

$$\mathbb{M} = (z_1)_1\mathbb{Z} \times \cdots (z_1)_d\mathbb{Z}$$

and

$$\mathbb{M}' = (z_2)_1\mathbb{Z} \times \cdots (z_2)_d\mathbb{Z}.$$

Consider now each pattern $p \in Q_2^{\mathcal{R}_{z_2}}$. Since the cylinder C_p defined by

$$C_p = \{c \in Q_2^{\mathbb{Z}^d} : \mathcal{P}_0^{z_2}(c) = p\}$$

is both open and closed, so is $\pi^{-1}(C_p)$. By compactity, and since cylinders form a basis of the topology, we get that $\pi^{-1}(C_p)$ is a finite union of cylinders of $Q_1^{\mathbb{Z}^d}$. We can suppose without loss of generality that they are all of shape $\mathcal{R}_{(2r+1)z_1}$ for some large enough r (finite unions of cylinders of small shape can always be defined as finite unions of cylinders of larger shapes). Doing this with the same value of r for all p , we get a (possibly partial) function f from $Q_1^{\mathcal{R}_{(2r+1)z_1}}$ to $Q_2^{\mathcal{R}_{z_2}}$. By eventually completing f and by definition of the factoring π between \mathbb{M} and \mathbb{M}' -scaled actions, f induces a local map from $Q_1^{\mathbb{Z}^d}$ to $Q_2^{\mathbb{Z}^d}$ associated with shapes \mathcal{R}_{z_1} and \mathcal{R}_{z_2} . It is onto because π is onto.

For cellular automata, the reasoning is similar and adding the temporal component in actions translates exactly into the desired property of weak commutation between the global maps of cellular automata and the onto map between configuration spaces. \square

Proof (Theorem 1). For the case of surjective CA, it is enough to notice that surjectivity is preserved by the relation \preceq . Indeed, if $F \preceq G$ we have

$$\phi \circ G^{t_1} = F^{t_2} \circ \phi$$

for some onto map ϕ . Therefore F must be surjective if G is surjective.

Then the proof follows from Kari's theorem [11] establishing that surjective CA are not recursively enumerable. Indeed, given a surjective universal CA U , we could enumerate thanks to the local presentation of factors (proposition 1) all CA F such that $F \preceq U$: they are all surjective (surjectivity is preserved by

factor) and all surjective CA are among them (universality).

We consider now the case of subshifts of finite type. Without loss of generality, any subshift of finite type can be presented as a subshift Σ_L where L is a finite set of patterns having all the same shape \mathcal{R}_z for some z . By Berger's theorem [1] the set of such L verifying that Σ_L is not empty can not be recursively enumerated. We show below that the existence of a universal subshift of finite type implies the existence of an algorithm of enumeration of all L of the form above such that Σ_L is not empty.

So suppose that there exists some universal subshift of finite type Σ_{L_U} where L_U is a set of Q_U -patterns of shape \mathcal{R}_{z_U} . Obviously, Σ_{L_U} must be non-empty. For any L and any pattern p of larger shape, we say that p is L -valid if it contains no occurrence of any pattern from L (occurrence requires that one shape is completely included into the other).

Let L be a set of Q -patterns of shape \mathcal{R}_z and ψ be a local map from $Q_U^{\mathbb{Z}^d}$ to $Q^{\mathbb{Z}^d}$ associated to shapes \mathcal{R}_{z_1} and \mathcal{R}_{z_2} . Consider the minimal shape \mathcal{R}_{z_+} containing both \mathcal{R}_z and \mathcal{R}_{2z_2} . Since ψ is local, one can check in finite time the following property called *validity property*: any pattern p of shape \mathcal{R}_{z_+} which has a L_U -valid preimage via ψ is L -valid (the size of preimages of finite patterns depends on the radius r associated to ψ but details don't matter here). By the definition of local maps and the hypothesis on shapes, this property implies that $\psi(\Sigma_{L_U}) \subseteq \Sigma_L$ and therefore $\Sigma_L \neq \emptyset$ (the choice of shape \mathcal{R}_{2z_2} ensures that validity is checked inside blocks of shape \mathcal{R}_{z_2} but also across the boundary between two such adjacent blocks).

It follows that we can recursively enumerate couples (L, ψ) having the property above. More precisely, maps ψ are enumerated via their local presentation (shapes, radius and local function). This way, we can enumerate a list of finite languages L such that Σ_L is not empty. To conclude the proof it is sufficient to show that all L such that $\Sigma_L \neq \emptyset$ are present in the list. Suppose by contradiction that some L over alphabet Q with $\Sigma_L \neq \emptyset$ is such that no local map from $Q_U^{\mathbb{Z}^d}$ to $Q^{\mathbb{Z}^d}$ verifies the validity property above. By universality of Σ_{L_U} , there exists a local map ψ sending Σ_{L_U} to Σ_L . Let r , \mathcal{R}_{z_1} and \mathcal{R}_{z_2} , and local function f , be the parameters associated to ψ . For any $k \geq 0$ we can define the same map ψ with another presentation by increasing artificially the radius r to kr and changing the local function f accordingly (shapes are kept unchanged). We call it the k^{th} presentation of ψ . Since, by hypothesis on L , no such presentation has the validity property, we deduce that there must exist some finite pattern p which is not L -valid and such that, for any k , p has a L_U -valid preimage under the k^{th} presentation of ψ . Therefore, by a simple compactness argument, there exists $c \in \Sigma_{L_U}$ such that $\psi(c)$ has an occurrence of p . Hence, $\psi(c) \notin \Sigma_L$ which is a contradiction. \square

B Proofs from section 3

Proof of lemma 1: We first build a CA that satisfies our lemma for $k = 2, h = 0$. Its state set will be made of one binary layer, and a signal layer. The behavior is simple: when q_s appears it generates a signal that will keep oscillating between the $\#$. When the signal is generated for the first time, it initialize the area, turning the first binary cell to 1 and the other one to 0s. Then, the signal keep moving from right to left and back between the $\#$. Each time it goes to the right, it turns one more binary cell to 1. And when the rightmost cell's binary layer is finally turned to 1 a new special signal is sent to the left which will generate the q_f .

If two or more signals crosses, one of them may survive. If one of them is initializing it will survive.

So, in $2.l$ steps of computation, the total number of 1s may be non increasing only in the following cases:

- if there is no signal at all
- if all cell's binary layer is already 1 and in this case a q_f was generated
- if an initialization signal has been sent.

In particular, if a q_f appears at some step, then in the previous $4.l$ steps, a 1 was generated. And in each previous $2.l$ step, at least a 1 was generated.

Thus, in the previous $2.l^2$ steps, at least one initialization signal was launched and a q_s has appeared. But by construction, after a q_s state appears, the first q_f state appears only exactly $2.l^2$ steps later.

It concludes the proof of the clock lemma in case $k = 2, h = 0$. For other values of k , simply slow down the signal going right to left. For other values of h , after the end of the quadratic part, launch a signal that will go right with speed 1 and come back left with speed $1/(h - 1)$ before raising q_f . \square

Proof of lemma 5:

By definition of a correct pattern, u is given by:

$$\# C_0 |Transition\ table| |State| |memory| \#$$

First of all, the $\#$ are never created or destroyed. The transition table and maximal state information are never modified, so they are the same in u and in v . And the sub-alphabet corresponding to control state, current state and memory alphabets are stable. The structure of v is the same as this of u . To prove our lemma it remains to prove that the second letter in v is C_0 , and that the current state value is smaller than the maximal value.

But, to make C_0 appear, at step t , a signal q_f was raised by the global clock, which implied, using the clock lemma, that the second letter of v is C_0 .

Thus all tests are launched. \square