CONSTRUCTION OF μ-LIMIT SETS

L. BOYER [1], M. DELACOURT [2], AND M. SABLIK [3]

[1] LAMA, Université de Savoie

[2] Laboratoire d'Informatique Fondamentale de Marseille, Université de Provence

[3] Laboratoire d'Analyse, Topologie, Probabilités, Université de Provence

Abstract. The μ-limit set of a cellular automaton is a subshift whose forbidden patterns are exactly those, whose probabilities tend to zero as time tends to infinity. In this article, for a given subshift in a large class of subshifts, we propose the construction of a cellular automaton which realizes this subshift as μ-limit set where μ is the uniform Bernoulli measure.

1. Introduction

A cellular automaton is a complex system defined by a local rule which acts synchronously and uniformly on the configuration space. These simple models have a wide variety of different dynamical behaviors. More particularly it is interesting to understand its behavior when it goes to infinity.

In the dynamical systems context, it is natural to study the limit set of a cellular automaton, it is the set of configurations that can appear arbitrarily far in time. This set captures the longterm behavior of the CA and has been widely studied since the end of the 1980s. Given a cellular automaton, it is difficult to determine its limit set. Indeed it is undecidable to know if it contains only one configuration [Kar92] and more generally, every nontrivial property of limit sets is undecidable [Kar94]. An other problem consists to characterize which subshift can be obtained as limit set of a cellular automaton. This was first studied in detail by Lyman Hurd [Hur87], and important progress has been made [Maa95, FK07] but there is still no characterization. The notion of limit set can be refined if we consider the notion of attractor [Hur90a, Kur03].

However, these topological notions do not correspond to the empirical point of view where the initial configuration is chosen randomly, that is to say chosen according a measure μ. That's why the notion of μ-attractor is introduced by [Hur90b]. Like it is discussed in [KM00] with a lot of examples, this notion is not satisfactory empirically and the authors introduce the notion of μ-limit set. A μ-limit set is a subshift whose forbidden patterns are exactly those, whose probabilities tend to zero as time tends to infinity. This set corresponds to the configurations which are observed when a random configuration is iterated.

Thanks to the project ANR EMC: ANR-09-BLAN-0164.

Submitted to JAC (Symposium on Cellular Automata Journées Automates Cellulaires)

As for limit sets, it is difficult to determine the μ-limit set of a given cellular automaton, indeed it is already undecidable to know if it contains only one configuration [BPT06]. However, in the literature, all μ-limit sets which can be found are very simple (transitive subshifts of finite type). In this article, for every enumerable family $(\Sigma_i)_{i\in\mathbb{N}}$ of subshifts generated by a generic configuration, we construct a cellular automaton which realizes $\overline{\bigcup_{i\in\mathbb{N}}\Sigma_i}$ as μ-limit set. In particular all transitive sofic subshifts can be realized. This show a strong difference with the limit set since there are sofic subshifts as the even subshift (subshift on alphabet $\{0,1\}$ in which all words 01^k0 with odd k are forbidden) which cannot be realized as limit set [Maa95].

To construct a cellular automaton that realizes a given subshift as μ-limit set, we begin to erase nearly all the information contained in a random configuration thanks to counters (section 3). Then we produce segments, which are finite areas of computation. On each segment we construct small parts of the generic configurations of many subshifts, and as time passes, segments grow larger and every word of every subshift appears often enough (section 5).

2. Definitions

2.1. Words and density

For a finite set Q called an *alphabet*, denote $Q^* = \bigcup_{n\in\mathbb{N}} Q$ the set of all finite words over Q. The *length* of $u = u_1u_2\dots u_n$ is $|u| = n$. We denote $Q^{\mathbb{Z}}$ the set of *configurations* over Q, which are mappings from $\mathbb{Z}$ to Q, and for $c \in Q^{\mathbb{Z}}$, we denote c_z the image of $z \in \mathbb{Z}$ by c. For $u \in Q^*$ and $0 < i \leq j \leq |u|-1$ we define the *subword* $u_{[i,j]} = u_iu_{i+1}\dots u_j$; this definition can be extended to a configuration $c \in Q^{\mathbb{Z}}$ as $c_{[i,j]} = c_ic_{i+1}\dots c_j$ for $i, j \in \mathbb{Z}$ with $i \leq j$. The *language* of $S \subset Q^{\mathbb{Z}}$ is defined by

$$\mathcal{L}(S) = \{u \in Q^* : \exists c \in Q^{\mathbb{Z}},\ \exists i \in \mathbb{Z} \text{ such that } u = c_{[i,i+|u|-1]}\}.$$

For every $u \in Q^*$ and $i \in \mathbb{Z}$, we define the *cylinder* $[u]_i$ as the set of configurations containing the word u in position i that is to say $[u]_i = \{c \in Q^{\mathbb{Z}} : c_{[i,i+|u|-1]} = u\}$. If the cylinder is at the position 0, we just denote it by $[u]$.

For all $u, v \in Q^*$ define $|u|_v$ the *number of occurences* of v in u as:

$$|u|_v = \text{card}\{i \in [0, |u| - |v|] : u_{[i,i+|v|-1]} = v\}$$

For a configuration $c \in Q^{\mathbb{Z}}$, the *density* $d_c(v)$ of a finite word v is:

$$d_c(v) = \limsup_{n\to+\infty} \frac{|c_{[-n,n]}|_v}{2n+1-|v|}.$$

These definitions could be generalized, for a set of words $W \subset Q^*$, we note $|u|_W$ and $d_c(W)$.

Definition 2.1 (Normal configuration)**.** A configuration is said to be *normal* for an alphabet Q if all words of length n have the same density of apparition in the configuration.

2.2. Subshifts

We denote σ the *shift* map $\sigma : Q^{\mathbb{Z}} \mapsto Q^{\mathbb{Z}}$ defined by $\sigma(c)_i = c_{i-1}$. A *subshift* is a closed, σ-invariant subset of $Q^{\mathbb{Z}}$. It is well known that a subshift is completely described by its language denoted $\mathcal{L}(\Sigma)$. Moreover, it is possible to define a subshift by a set of its forbidden words which do not appear in the language.

As the shift invariance is preserved, intersections and closures of unions of subshifts are still subshifts. And in particular, the union of a set $(\mathcal{L}(\Sigma_i))_i$ of languages describes the subshift that is the closure of the union of all subshifts: $\overline{\bigcup_{i\in\mathbb{N}} \Sigma_i}$.

We are going to define some class of subshift. A *sofic subshift* is a subshift whose language of forbidden words is rational, i.e. given by a finite automaton. A subshift Σ is *transitive* if for all $u, v \in \mathcal{L}(\Sigma)$ there exists a word w such that $uwv \in \mathcal{L}(\Sigma)$. Let $s : Q \to Q^*$ be a primitive substitution (there exists $k \in \mathbb{N}$ such that for all $a, b \in Q$ a appears in $s^k(b)$), the *substitive subshift* associated to s is the subshift Σ_s such that

$$\mathcal{L}(\Sigma_s) = \{u \in Q^* : \exists a \in Q \text{ and } n \in \mathbb{N} \text{ such that } u \text{ appears in } s^n(a)\}.$$

2.3. Cellular automata

Definition 2.2 (Cellular automaton)**.** A *cellular automaton (CA)* $\mathcal{A}$ is a triple $(Q_\mathcal{A}, r_\mathcal{A}, \delta_\mathcal{A})$ where $Q_\mathcal{A}$ is a finite set of states called the *alphabet*, $r_\mathcal{A}$ is the *radius* of the automaton, and $\delta_\mathcal{A} : Q_\mathcal{A}^{2r+1} \mapsto Q_\mathcal{A}$ is the *local rule*.

The configurations of a cellular automaton are the configurations over $Q_\mathcal{A}$. A global behavior is induced and we'll note $\mathcal{A}(c)$ the image of a configuration c given by: $\forall z \in \mathbb{Z}, \mathcal{A}(c)_z = \delta_\mathcal{A}(c_{z-r}, \dots, c_z, \dots, c_{z+r})$. Studying the dynamic of $\mathcal{A}$ is studying the iterations of a configuration by the map $\mathcal{A} : Q_\mathcal{A}^{\mathbb{Z}} \to Q_\mathcal{A}^{\mathbb{Z}}$. When there is no ambiguity, we'll note Q, r and δ for $Q_\mathcal{A}$, $r_\mathcal{A}$, $\delta_\mathcal{A}$.

2.4. μ-limit sets

Definition 2.3 (Uniform Bernoulli measure)**.** For an alphabet Q, the *uniform Bernoulli measure* μ on configurations over Q is defined by: $\forall u \in Q^*, i \in \mathbb{Z}, \mu([u]_i) = \frac{1}{|Q|^{|u|}}$.

For a CA $\mathcal{A} = (Q, r, \delta)$ and $u \in Q^*$, we denote for all $n \in \mathbb{N}$, $\mathcal{A}^n\mu([u]) = \mu\left(\mathcal{A}^{-n}([u])\right)$.

Definition 2.4 (Persistent set)**.** For a CA $\mathcal{A}$, and the uniform Bernoulli measure μ, we define the *persistent set* $L_\mu(\mathcal{A})$ with: $\forall u \in Q^*$:

$$u \notin L_\mu(\mathcal{A}) \iff \lim_{n\to\infty} \mathcal{A}^n\mu([u]_0) = 0.$$

Then the *μ-limit set* of $\mathcal{A}$ is $\Lambda_\mu(\mathcal{A}) = \left\{c \in Q^{\mathbb{Z}} : L(c) \subseteq L_\mu(\mathcal{A})\right\}$.

Remark 2.5. As this definition gives a set of forbidden finite words, we clearly see that μ-limit sets are subshifts.

Definition 2.6 (Set of predecessors)**.** We define the set of predecessors at time n of a finite word u for a CA $\mathcal{A} = (Q, r, \delta)$ as $P_\mathcal{A}^n(u) = \left\{v \in Q^{|u|+2rn} : \mathcal{A}^n([v]_{-rn}) \subseteq [u]_0\right\}$.

Remark 2.7. As we consider the uniform Bernoulli measure μ, $\frac{|P^n_{\mathcal{A}}(u)|}{|Q|^{|u|+2rn}} \to 0 \Leftrightarrow u \notin L_\mu(\mathcal{A})$.

Remark 2.8. The set of normal configurations has measure 1 in $Q^{\mathbb{Z}}$. Which means that a configuration that is randomly generated according to measure μ is a normal configuration.

Lemma 2.9. *Given a CA $\mathcal{A}$ and a finite word u, with μ the uniform Bernoulli measure, for any normal configuration c:*
$u \in \Lambda_\mu(\mathcal{A}). \Leftrightarrow d_{\mathcal{A}^n(c)}(u) \nrightarrow 0$ when $n \to +\infty$.

Proof. Actually, we prove here that for the uniform measure and for any $n \in \mathbb{N}$:
$d_{\mathcal{A}^n(c)}(u) = \mathcal{A}^n\mu(u) = \frac{|P^n_{\mathcal{A}}(u)|}{|Q|^{|u|+2rn}}$.
The second equality is clear.
Let $n \in \mathbb{N}$. We note r the radius of $\mathcal{A}$. Since any occurence of u in $\mathcal{A}^n(c)$ corresponds to an occurence of a predecessor of u in c :

$$d_{\mathcal{A}^n(c)}(u) = \limsup_{k\to+\infty} \frac{|\mathcal{A}^n(c)_{[-k,k]}|_u}{2k+1-|u|} = \limsup_{k\to+\infty} \sum_{v\in P^n_{\mathcal{A}}(u)} \frac{|c_{[-k-rn,k+rn]}|_v}{2k+2rn+1-(|u|+2rn)}.$$

And as c is normal, for any $v \in P^n_{\mathcal{A}}(u)$: $|c_{[-k-rn,k+rn]}|_v \sim_{k\to+\infty} \frac{2k+1}{|Q|^{|u|+2rn}}$.
Then:

$$d_{\mathcal{A}^n(c)}(u) = \sum_{v\in P^n_{\mathcal{A}}(u)} \limsup_{k\to+\infty} \left(\frac{1}{2k+1-|u|}\frac{2k+1}{|Q|^{|u|+2rn}}\right) = \sum_{v\in P^n_{\mathcal{A}}(u)} \frac{1}{|Q|^{|u|+2rn}} = \frac{|P^n_{\mathcal{A}}(u)|}{|Q|^{|u|+2rn}}.$$

∎

Proposition 2.10. *Let $u \in L_\mu(\mathcal{A})$, there exists a word w such that $uwu \in L_\mu(\mathcal{A})$.*

Proof. Let $u \in L_\mu(\mathcal{A})$, there exists $\alpha > 0$ and an increasing sequence $(n_i)_{i\in\mathbb{N}}$ such that $\mathcal{A}^{n_i}\mu([u]) > \alpha$. Thus, for a normal configuration c, one has $d_{\mathcal{A}^{n_i}(c)}(u) > \alpha$ for all $i \in \mathbb{N}$. Let $l \in \mathbb{N}$ and $\epsilon > 0$ such that $\frac{2|u|}{2|u|+l} < \alpha - \epsilon$, we define

$$\begin{aligned} W_1 &= \{w \in Q_A^* : u \text{ is not a subword of } w \text{ and } |w| \le l\} \text{ and} \\ W_2 &= \{w \in Q_A^* : u \text{ is not a subword of } w \text{ and } |w| > l\}. \end{aligned}$$

Consider $uW_ku = \{uwu : w \in W_i\}$ for $k \in \{1,2\}$, one has

$$d_{\mathcal{A}^{n_i}(c)}(uW_2u) = \limsup_{n\to\infty} \frac{|\mathcal{A}^{n_i}(c)_{[-n,n]}|_{uW_2u}}{2n+1} \le \frac{2|u|}{2|u|+l}$$

since a word of uW_2u can appear at most $2|u|$ times for each pattern of length $2|u|+l$ of $\mathcal{A}^{n_i}(c)$. Moreover $d_{\mathcal{A}^{n_i}(c)}(uW_1u)+d_{\mathcal{A}^{n_i}(c)}(uW_2u) \ge d_{\mathcal{A}^{n_i}(c)}(u)$ so $d_{\mathcal{A}^{n_i}(c)}(uW_1u) \ge \epsilon$.

Since W_1 is finite, there exists a word $w \in W_1$ such that $d_{\mathcal{A}^{n_i}(c)}(uwu) \ge \epsilon$ for an infinity of $i \in \mathbb{N}$. Thus $uwu \in L_\mu(\mathcal{A})$. ∎

Example 2.11. We consider here the "max" automaton $\mathcal{A}_M$. The alphabet contains only two states 0 and 1. The radius is 1. When the rule applies to three 0 (no 1), it produces a 0. In any other case, it produces a 1.

The probability to have a 0 at time t is the probability to have 0^{2t+1} on the initial configuration. Which tends to 0 when $t \to \infty$ for the uniform Bernoulli measure. So, 0 does not appear in the μ-limit set. And finally $\Lambda_\mu(\mathcal{A}_M) = \{^\infty 1^\infty\}$.

And this example gives a difference between subshifts that can be realised as limit set ($\Lambda(\mathcal{A}) = \bigcap_{i\in\mathbb{N}} \mathcal{A}^i(Q^{\mathbb{Z}})$) and subshifts that can be realised as μ-limit set.

Effectively, $\Lambda(\mathcal{A}_M) = ({}^\infty 10^* 1^\infty) \bigcup ({}^\infty 0^\infty) \bigcup ({}^\infty 01^\infty) \bigcup ({}^\infty 10^\infty)$, but if we apply proposition 2.10 with the word 01, we conclude that $\Lambda(\mathcal{A}_M)$ cannot be a μ limit set.

3. Counters

In this section and the following one, we describe an automaton $\mathcal{A}_S$, which, on normal configurations, produces finite segments of size growing with time. In these segments, we will make computations described in section 5.

Before to start computation, the automaton $\mathcal{A}_S$ has a transitory regime which erases the random configuration and generate segments between # where the computation is done. To do that, we have a special state $*$, that can only appear in the initial configuration, and which generates two counters. Between two counters, the states are initialized and when two counters intersect, they compare their respective lengths. If they do not have the same age, the younger deletes the older one; if they have the same age, they disappear and we put the state # in view to start the computation. The notion of counters was introduced in [DPST10] to produce equicontinuous points according arbitrary curves.

We recall some ideas which allow to construct such automaton:

- no transition rule produces the state $*$;
- $*$ produces two couples of signals, one toward the left and another one toward the right;
- a couple of signal (called *counter*) is formed by an *inner* signal and an *outer* signal, which is faster. Their collisions are handled in the following way:
  - nothing other than an outer signal can go through another outer signal;
  - when two outer signals collide they move through each other and comparison signals are generated;
  - on each side, a signal moves at maximal speed towards the inner border of the counter, bounces on it and goes back to the point of collision;
  - the first signal to come back is the one from the youngest counter and it then moves back to the outer side of the oldest counter and deletes it;
  - the comparison signal from the older counter that arrives afterwards is deleted and will not delete the younger counter's outer border;
- between a left counter and a right counter, the configuration is initialized;
- if two counters that have the same age meet, they disappear and produce the state $\#_S$ which start the computation described in section 5
- the state $\#_S$ becomes # which delimitates segments, this state can disappear if two adjacent segments decide to merge as described in section 4, or if a counter (necessarily younger) encounters it.

The initialization of a configuration is illustrated in figure 1. The gray areas of computation begin on the left of a # produced by the meeting of two counters generated by a $*$.

Lemma 3.1. *There exists a constant K_c such that if two # are distant of k, they appeared before time $k \times K_c$.*

Proof. Consider two states # in the space time diagram separated by k cells. If # is not in the initial configuration, the only way to appear is to result from the collision of two counters coming from the left and from the right. Thus, in the initial

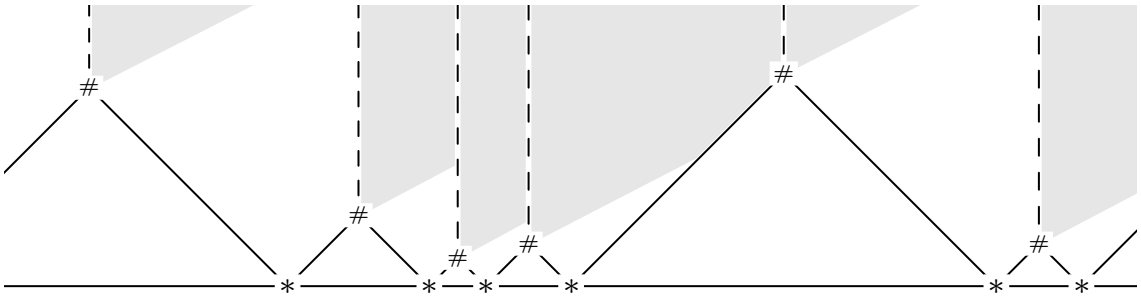

Figure 1: When two counters launched by a $*$ meet, a # is produced and a computation is launched on the right. The computation area extends until it meets the inner signal of a counter or another #.

configuration, it is necessary to have the state $*$ between the two # to create the two #. This operation take at most $k \times K_c$ where K_c is the speed of an inner signal. ∎

4. Merging segments

We saw in Section 3, how a special state $*$ on the initial configuration gave birth to counters protecting everything inside them until they meet some other counter born the same way. In this section, we will describe the evolution of the automaton $\mathcal{A}_S$ after this time of initialization. When two counters of the same age meet, they disappear and a # is produced.

Definition 4.1 (Segment)**.** A *segment* u is a subword of a configuration delimited by two # and containing no # inside. So, $u \in \#\left(Q \setminus \{\#\}\right)^*\#$. The *size* of a segment is the number of cells between both #.

There will be computations made inside segments, but we will describe it later. Thus, in a segment, there is a layer left for computations that remain inside the both #, and a "merging layer" that will contain signals necessary for the behavior with other segments. Every signal presented in this section will travel on this merging layer. The idea is the following: at some times, two neighbor segments will decide to merge together to form one unique segment whose size will be the sum of both sizes. And we will assure that each segment will eventually merge, so that no segment of finite size can still be in the μ-limit set of $\mathcal{A}_S$.

When a # is produced in automaton $\mathcal{A}_S$, it sends two signals, on its right and on its left to detect the first # on each side. If the signal catches the inside of a counter still in activity before reaching a #, it waits until the counter produces a #. Then both # have recognised each other and the segment between them is "conscious". It launches a computation inside it, and waits until it is achieved. We will assure later that this computation ends. When this is done, it will alternatively send signals on its left and on its right to propose successively to its neighbors to merge.

For this purpose, it computes and stores the length n of the segment as a binary representation. Then the segment puts a L mark on the # to its left, and waits for n^2 timesteps. If, during this time, the left side neighbor has not put a R mark on the common #, our segment erases the L mark, a signal is sent on the other side,

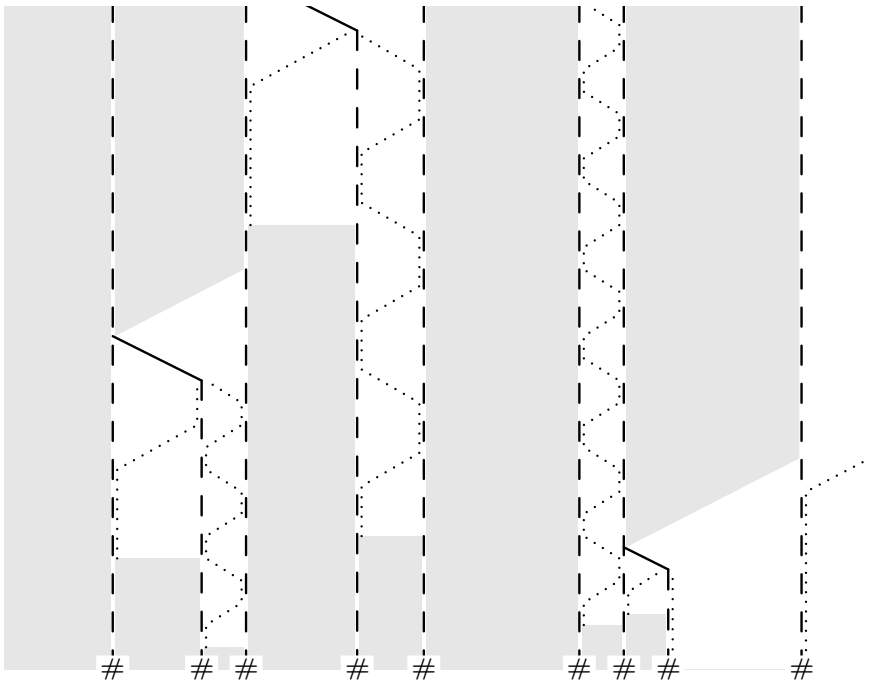

Figure 2: A # stays until two segments merge. Computation happens in gray areas, and at its end a signal (...) is sent and stays on the left of the segment, then goes to the right, stays and comes back. This cycle continues until a neighbor's signal is on the same # at the same time. Then the # is deleted and another computation is started on the left.

and it puts a R mark on the # to its right end. It waits once again n^2 timesteps before erasing the R, sending a signal to its left, and starting over. The whole cycle takes $2(n^2 + n)$ timesteps as we consider a signal at speed 1 crossing a segment of size n. We request the signal to stay n^2 timesteps because as $(n + 1)^2 > n^2 + 2n$, if two segments do not have the same size, their signals eventually meet during a cycle of the smallest one. So, the only case in which two neighbor segments that try to merge do not merge, is when they have same size and are correctly synchronized. Computing and storing n, and waiting n^2 can be done with a space $\log(n)$.

This process ends when at the same time, both a L and a R mark are written on a #. When this happens, the two segments agree to merge together and they do it. Which means, they erase every data on the merging layer, the # between them is erased too, and the whole activity begins again, starting with the computation inside the new segment.

The general behavior of the segments among themselves is illustrated in figure 2. We prove the following claims for automaton $\mathcal{A}_S$.

Claim 4.2. *For any two words $u, v \in Q^*$, with $|u| \neq |v|$, if the word $w = \#u\#v\#$ appears at time t in a space-time diagram of $\mathcal{A}_S$, one of the 3 # of w has disappeared at time $t + |Q|^{|w|} + 2(|w|^2 + |w|)$.*

Proof. If the word w exists at time t on a space time diagram, at time $t + |Q|^{|u|}$ (respectively $t + |Q|^{|v|}$) at most, the computation is achieved in u (resp. v). We

suppose here that no # in w has disappeared, which means, u and v do not merge with any other segment outside w. So at time $t+|Q|^{|w|}$ both segments try to merge with another one. Assume $|v|>|u|$ for example, the other case is totally symmetric. Then, as $|v|^2>|u|^2+2|u|$, before the end of the cycle of v, they have put their mark simultaneously on their common # for one timestep at least. And consequently, they have merged and one # has disappeared at time $t+|Q|^{|w|}+2(|w|^2+|w|)$. ∎

Claim 4.3. *Two segments of size less than $k\in\mathbb{N}$ can merge together at most $|Q|^{2k}+2((2k)^2+2k)$ timesteps after being formed.*

Proof. If they don't have the same size, lemma 4.2 let us conclude. If they have the same size, their computations are achieved after $|Q|^{2k}$. And as their merging cycle takes the same time for both, if they do not merge during the first cycle, they will never merge. So they merge before $|Q|^{2k}+2((2k)^2+2k)$ too.

∎

Claim 4.4. *For any two words $u,v\in Q^*$, with $|u|\neq|v|$, the word $w=\#u\#v\#$ does not appear in $\Lambda_\mu(\mathcal{A}_S)$.*

Proof. We use the constant K_c from lemma 3.1. Denote $T=|w|\times K_c+|w|\left(|Q|^{|w|}+2(|w|^2+|w|)\right)$. We prove that for $t>T$, $P^t_{\mathcal{A}_S}(w)=\emptyset$. If the two # encircling w never disappear, the dynamic inside w is not affected by the exterior. Through time, some other # possibly appeared and disappeared between them. But after time at most $|w|\times K_c$, they have all appeared. Since then, they will only disappear. There are less than $|w|-1$ excedentary # that have to disappear. Considering lemma 4.3, one disappears at least every $|Q|^{|w|}+2(|w|^2+|w|)$ timesteps. After that, the two segments of w are formed, and with lemma 4.2, one of the # of w disappear before $|Q|^{|w|}+2(|w|^2+|w|)$ new timesteps. Finally, at time T, one of the # of w has disappeared and $P^t_{\mathcal{A}_S}(w)=\emptyset$. ∎

Proposition 4.5. *There is no # in the μ-limit set of $\mathcal{A}_S$.*

Proof. Assume that $\#\in L_\mu(\mathcal{A})$, by Proposition 2.10, there exits $u\in Q^*$ such that $\#u\#\in L_\mu(\mathcal{A})$, we can assume that u does not contain #. Let $k=|u|$, by Lemma 3.1, the # encircling u appeared before time $k\times K_c$. Denote $W=\{\#v\#:v\in(Q\backslash\{\#\})^k\}$ and $X_n=\{x\in Q^{\mathbb{Z}}:\mathcal{A}^k(x)_{[0,k+1]}\in W \text{ for all } k\in[k\times K_c,n]\}$. Since $\#u\#\in L_\mu(\mathcal{A})$, there exists $\alpha>0$ such that $\mu(X_n)>\alpha$ for an infinity of $n\in\mathbb{N}$. Moreover, as $X_{n+1}\subset X_n$, we can conclude that $\mu(X_\infty)>\alpha$ where $X_\infty=\cap_{n\in\infty}X_N$.

As μ is Bernoulli, we have $\mu(Y)>0$ where $Y=[*(Q\setminus *)^{2k}*(Q\setminus *)^{2k}*]_0$; moreover there exist $k_1\geq 0$ and $k_2\geq k_1+4k+1$ such that $\mu(Z)>0$ where $Z=X_\infty\cap\sigma^{-k_1}(Y)\cap\sigma^{-k_2}(X_\infty)$. For all $n\geq k\times K_c$ one has $F^n(Z)_{[0,k+1]}\subset W$, $F^n(Z)_{[k_2,k_2+k+1]}\subset W$ and $F^n(Z)_{[k_1+k,k_1+3k]}\subset F^n(Y)_{[k_1+k,k_1+3k]}$ does not contain #.

We deduce that there exists a word $w\in Q$ of length k_2-k-1 such that $w_{[k_1+k,k_1+3k]}$ does not contain # and $\#u\#w\#u\#\in L_\mu(\mathcal{A})$. However, in $\#u\#w\#u\#$ we can find two segments $\#u_1\#u_2\#$ which have different length. By Claim 4.4 we obtain a contradiction. Thus, there is no # in the μ-limit set of $\mathcal{A}_S$. ∎

Finally, we prove a lemma that will be useful later.

Claim 4.6. *The density of cells outside segments generated by counters born in the initial configuration tends to 0.*

Proof. The proof is clear since such a cell needs predecessors without states $*$ on each side in the initial configuration. ∎

Lemma 4.7. *For a word $u \in Q^*$ of density less than α_k in segments of size greater than k, if $\alpha_k \to 0$ when $k \to \infty$, then $u \notin L_\mu(\mathcal{A}_S)$.*
Conversely, for a word $u \in Q^*$ of density more than α_k in segments of size greater than k, if $\alpha_k \nrightarrow 0$ when $k \to \infty$, then $u \in L_\mu(\mathcal{A}_S)$.

Proof. Let's consider a normal configuration c. For any $k \in \mathbb{N}$, we denote

$$d_k^t = \sum_{v \in \#(Q_\mathcal{A})^l\#,\ l \leq k} l \times d_{\mathcal{A}_S^t(c)}(v)$$

the density of cells in segments of size less than k in the image at time t of c. Due to proposition 4.5, $d_k^t \to 0$ when $t \to \infty$. And due to claim 4.6, the density a^t of cells outside wellformed segments tends to 0 when $t \to \infty$.

Suppose $u \in Q^*$ has density less than α_k in segments of size greater than k with $\alpha_k \to 0$. Concerning segments of size less than k, the density of u is then less than d_k^t, and for segments of size more than k, the density is less than α_k. Finally, at a given time t, $d_{\mathcal{A}_S^t(c)}(u) \leq d_k^t + \alpha_k + a^t$.

As this equation holds for any k, finally, when $t \to \infty$, $d_{\mathcal{A}_S^t(c)}(u)$ has a limit which is 0. This concludes the proof of the first part of the lemma with lemma 2.9.

In the other side, suppose $u \in Q^*$ has density more than α_k in segments of size k with $\alpha_k \nrightarrow 0$. Therefore, $d_{\mathcal{A}_S^t(c)}(u) \geq (1 - d_k^t - a^t)\alpha_k$ which does not tend to 0 when $t \to \infty$ and $k \to \infty$. Thus, $u \in L_\mu(\mathcal{A}_S)$.

■

5. Infinite Unions

In this section we will see how to create a cellular automaton whose μ-limit set is the closure of the infinite union of a recursively enumerable family of subshifts.

Definition 5.1 (Generable Subshift)**.** We say that a Turing machine M *generates* a subshift $\Sigma \subseteq Q^{\mathbb{Z}}$ if M computes a generic configuration of Σ in the following sense:

- the tape alphabet of M contains Q;
- on an empty tape, M writes the right half of a configuration $c \in \Sigma$ such that $\limsup_{n\to\infty} \frac{|c_{[0,n]}|_u}{n+1} > 0$ if and only if $u \in \mathcal{L}(\Sigma)$; c is called a *generic configuration*;
- after a symbol of Q has been written on the tape, it is never changed.

Theorem 5.2. *Given a recursively enumerable family $(\Sigma_i)_{i\in\mathbb{N}}$ of generable subshifts, that is to say that there exists a Turing machine that enumerates a set of machines $(M_i)_i \in \mathbb{N}$ such that M_i produces the subshift Σ_i, there exists a cellular automaton whose μ-limit set is exactly the subshift $\overline{\bigcup_{i\in\mathbb{N}} \Sigma_i}$.*

Proof. Let us consider a recursively enumerable family $(\Sigma_i)_{i\in\mathbb{N}}$ of generable subshifts, let us denote by M the Turing machine that enumerates the machines $(M_i)_{i\in\mathbb{N}}$ that in turn produce the generic configurations for each of the subshifts.

We will now describe the behavior of such a cellular automaton $\mathcal{A}$. $\mathcal{A}$ will work as the automaton $\mathcal{A}_S$ described in Section 4 meaning that from a normal configuration, it will generate "counter signals" that will produce finite segments on the configuration (separated by a # symbol). We will now describe the computation performed by each finite segment during the evolution of the cellular automaton.

The first thing a segment does is compute its length n and store it as a binary representation. By incrementing a binary counter moving across the segment, this is easily done in space $\log(n)$. Once this is done, the segment can simulate Turing machines on its first $\log(n)$ cells (it is important to limit the computational space so that the computation states become negligible and disappear from the μ-limit set).

On the initial $\log(n)$ cells of the segment the machine M is simulated so that it produces as many of the descriptions of the machines $(M_i)_{i\in\mathbb{N}}$ as possible, and stops when it has reached its space limit. Let us assume the machine has produced k descriptions (k may be 0 for short segments, but we know that as the segments grow larger, k will grow too). Additionnally, we request that $k \leq \log(\log(n))$.

The space of size $\log(n)$ is further divided into k fragments of size $\log(n)/k$. On each of these k fragments, the corresponding machine M_i is simulated to produce the first letters w_i of the generic configuration corresponding to the subshift Σ_i. The word w_i might be much smaller than $\log(n)/k$ depending on the space needed by the machine M_i to compute the first letters of its generic configuration, but again we know that as segments grow larger, larger words will be computed.

After the k different w_i have been computed, the initial segment of length n is split into $\sqrt{n}$ fragments of length $\sqrt{n}$. Each of these fragments is filled with copies of one of the w_i in the following manner: one out of two is filled with w_1, one out of four (i.e. one out of two among the remaining fragments) is filled with w_2, one out of eight is filled with w_3 and so on. The remaining segments (if k is very small, we might run out of w_i before filling all the fragments) are filled with w_k. Fragments are separated by a symbol $\$_1 \notin Q$ and the copies of words w_i inside a given fragment are separated by a symbol $\$_2 \notin Q$.

Remark 5.3. The previous construction can all be done using only $\log(n)$ cells of computation at a given time (cells that are not active and that only contain a symbol from $Q \cup \{\$_1, \$_2\}$ are not counted). To fill the fragments of size $\sqrt{n}$ we need only compute the binary expression of $\sqrt{n}$ and then advance through the segment while filling the fragment with the appropriate w_i while decreasing a counter to measure $\sqrt{n}$ cells. The important data (the words w_i and different counters) are moved through the segment so that they are always present near the location to be filled. Thus the head of the Turing machine M carries only $\log(n)$ cells used to store the w_i and to its computation. No mark of the computation remains in the other cells, even those already visited and rewritten.

When all the fragments of the segment have been filled with the w_i, the segment can erase all the remaining computation data and start the process of merging with its neighbors as described in Section 4.

When two segments merge, the whole computation is restarted but this time with a larger space. The segments are not erased immediately after a merge, but rather the new data overwrites the previous as the $\sqrt{n}$ fragments are filled.

We will prove that $L_\mu(\mathcal{A}) = \bigcup_{i\in\mathbb{N}} \mathcal{L}(\Sigma_i)$.

Claim 5.4. *The states used for computation, signals inside segments, writing fragments,* $\$_1$ *or* $\$_2$ *do not appear in* $\Lambda_\mu(\mathcal{A})$.

Proof. Here we use the lemma 4.7 for each of these states.

We use the $\log(k)$ initial cells of a segment of size k to do the computation, so the density of these cells is $\log(k)/k$, and the property is proved. The head of the Turing machine M carries at most $\log(k)$ cells for its computation or writing, thus

the same argument works. The signals for the merging process are in a finite number in a segment, therefore their density in a segment tends to 0 too. The density of $\$_1$ is $\sqrt(k)/k$, and the lemma applies once again.

For the density of $\$_2$, let $\lambda > 0$, $\exists k_0 > 0$ such that the word w_i produced in a segment of size $k > k_0$ is such that $|w_i| > \lambda$ for any $i \leq \lambda$. So, for $k > k_0$, the density of $\$_2$ in a segment of size k is less than $1/\lambda$ in fragments of $S_i, i \leq \lambda$ and less than 1 in the other fragments that have themselves a density lower than $1/2^\lambda$. And thus, the density of $\$_2$ is lower than $\frac{1}{\lambda} + \frac{1}{2^\lambda}$ in segments of size $k > k_0$. Finally the density of $\$_2$ tends to 0 when $k \to \infty$. And the claim is proved. ∎

Claim 5.5. *For any subshift Σ_i, $i \in \mathbb{N}$, any word $u \in \mathcal{L}(\Sigma_i)$ and any family of segments $(v_k)_k$ of size $|v_k| = k$, $d_{v_k}(u)$ does not tend to 0 when $k \to \infty$.*

Proof. As $u \in \mathcal{L}(\Sigma_i)$, its density $\alpha(u)$ in the generic configuration computed by M_i is positive. So, there exists $l_i \in \mathbb{N}$ such that any subword of this configuration contains u with density at least $\alpha(u)/2$. Let k_0 such that in any segment of size $k > k_0$, the word w_i computed has length $|w_i| > l_i$.

For any segment v_k of size $k > k_0$, there are $\log(k)$ cells occupied for computation, less than $\sqrt(k)$ cells containing a $\$_1$ and $\frac{1}{2^{i+1}}$ among the remaining cells attributed to the copies of w_i. Among these copies, a proportion $\frac{l_i-1}{l_i}$ of the cells contain $\$_2$. $\log(k)$ additional cells can be dedicated to the head of the Turing machine M writing in the segment and a finite number K of cells can contain signals for the merging process. Finally,

$$d_{v_k}(u) \geq \left(\left(\frac{k - \log(k) - \sqrt(k)}{2^{i+1}} \right) \frac{l_i - 1}{l_i} - \log(k) - K \right) \frac{1}{k}.$$

Which does not tend to 0 when $k \to \infty$. ∎

Claim 5.6. *For any subshift Σ_i, $i \in \mathbb{N}$ and any word $u \in \mathcal{L}(\Sigma_i)$, $u \in L_\mu(\mathcal{A})$.*

Proof. We clearly get the result by combining claim 5.5 and lemma 4.7. ∎

Finally, the theorem is proven:

- the proposition 4.5 and the claim 5.4 assure that every state used for computation does not appear in $\Lambda_\mu(\mathcal{A})$, which means $L_\mu(\mathcal{A}) \subseteq \bigcup_{i \in \mathbb{N}} \mathcal{L}(\Sigma_i)$,
- the claim 5.6 assures that $\bigcup_{i \in \mathbb{N}} \mathcal{L}(\Sigma_i) \subseteq L_\mu(\mathcal{A})$.

∎

The next proposition gives some examples of generable subshifts.

Proposition 5.7. *The following subshifts are generable:*

- *transitive sofic subshifts,*
- *substitutive subshift associated to a primitive substitution.*

Proof. As a transitive sofic subshift Σ is given by the strongly connected automaton recognizing its language. For example, we can write successively every cycle of size k for k from 1 to ∞. In this case we obtain a configuration where the density of all the words of the language of Σ is positive.

For a primitive substitution s, it is easy to generate the fix point configuration denoted $c_{[0;\infty]}$ whose all prefixes are given by $s^k(a)$ for all $k \in \mathbb{N}$ where $a \in Q$. It

is well know that all words of the substitutive subshift associated appears with a positive density in $c_{[0;\infty]}$ [Fog05]. ∎

6. Conclusion and perspectives

In this paper, we prove that a large class of subshifts can be realized as μ-limit sets of cellular automata. In particular, it is possible to obtain all transitive sofic subshifts, this is a profound difference with the topological case since the even shift cannot be realized as the limit set of one cellular automaton. This construction allows to control the iterations of a random configuration in view to obtain an auto-organized behavior. The construction can be adapted at least in two ways:

- to obtain the same result for a large class of measure (σ-ergodic measure of full support) modulo some technical changes
- to obtain a subshift without any word of low complexity (as suggested by V. Poupet).

Of course the principal open question is in the reciprocal of the theorem, that is to say to characterize subshifts that can possibly be realized as μ-limit sets.

Acknowledgments

We are deeply grateful to Victor Poupet and Guillaume Theyssier for their ideas, and constant support to the writing of this article.

References

[BPT06] Laurent Boyer, Victor Poupet, and Guillaume Theyssier. On the Complexity of Limit Sets of Cellular Automata Associated with Probability Measures. *MFCS 2006*, LNCS 4162:190–201, 2006.

[DPST10] Martin Delacourt, Victor Poupet, Mathieu Sablik, and Guillaume Theyssier. Directional Dynamics along Arbitrary Curves in Cellular Automata. *Theoretical Computer Science*, A paraître, 2010.

[Fog05] N. Pytheas Fogg. Substitutions in Dynamics, Arithmetics and Combinatorics. V. Berthé, S. Ferenczi, C. Mauduit, A. Siegel (Eds), 2005.

[FK07] Enrico Formenti and Petr Kurka. A Search Algorithm for the Maximal Attractor of a Cellular Automaton. *STACS, 2007*, pages 356–366, 2007.

[Hur87] Lyman P. Hurd. Formal Language Characterizations of Cellular Automata Limit Sets. *Complex Systems*, 1:69–80, 1987.

[Hur90a] Mike Hurley. Attractors in cellular automata. *Ergodic Theory Dynam. Systems*, 10(1):131–140, 1990.

[Hur90b] Mike Hurley. Ergodic aspects of cellular automata. *Ergodic Theory Dynam. Systems*, 10(4):671–685, 1990.

[Kar92] Jarkko Kari. The Nilpotency Problem of One-Dimensional Cellular Automata. *SIAM J. Comput.*, 21(3):571–586, 1992.

[Kar94] Jarkko Kari. Rice's Theorem for the Limit Sets of Cellular Automata. *Theor. Comput. Sci.*, 127(2):229–254, 1994.

[KM00] Petr Kurka and Alejandro Maass. Limit sets of cellular automata associated to probability measures. *Journal of Statistical Physics*, 100(5):1031–1047, 2000.

[Kur03] Petr Kurka. *Topological and symbolic dynamics*. Société Mathématique de France, Paris, 2003.

[Maa95] Alejandro Maass. On the sofic limit sets of cellular automata. *Ergodic Theory Dynam. Systems*, 15:663–684, 1995.