Submitted to the Symposium on Theoretical Aspects of Computer Science
www.stacs-conf.org

AN ORDER ON SETS OF TILINGS CORRESPONDING TO AN ORDER ON LANGUAGES

NATHALIE AUBRUN [1] AND MATHIEU SABLIK [2]

[1] Institut Gaspard Monge, Université Paris-Est Marne-la-Vallée,
77454 Marne-la-Valle Cedex 2, France.
E-mail address: `nathalie.aubrun@ens-lyon.fr`

[2] Laboratoire d'Analyse, Topologie, Probabilité, Université de Provence,
39, rue F. Joliot Curie, 13453 Marseille Cedex 13, France.
E-mail address: `sablik@cmi.univ-mrs.fr`

Abstract. Traditionally a tiling is defined with a finite number of finite forbidden patterns. We can generalize this notion considering any set of patterns. Generalized tilings defined in this way can be studied with a dynamical point of view, leading to the notion of subshift. In this article we establish a correspondence between an order on subshifts based on dynamical transformations on them and an order on languages of forbidden patterns based on computability properties.

Introduction

Given a finite set of tiles $\mathcal{A}$ and a finite set of forbidden patterns P, a d-dimensional tiling is an element of $\mathcal{A}^{\mathbb{Z}^d}$ where the local conditions imposed by P are satisfied at every point of $\mathbb{Z}^d$. This basic model captures geometrical aspect of computation [Ber66, **?**, Han74]. To estabish structural properties of tilings, it is interesting to study the set of tilings which satisfies the conditions imposed by P [BDJ08].

It is easy to generalize the usual notion of tiling considering infinite set of forbidden patterns. A set of generalized tilings can be studied with a dynamical point of view with the notion of subshift [LM95, Kit98]. In this theory, a set of usual tilings corresponds to a subshift of finite type.

In dimension 1, the class of subshifts of finite type is well understood. In particular, the language of a subshift of finite type is given by a local automaton [Bea93]. Given this result, it is natural to characterize subshifts which of language is given by a finite automaton. It is the class of sofic subshifts which can all be obtained as a factor of a subshift of finite type [LM95]. Thus, each sofic subshift is obtained by a dynamical transformation of a subshift of finite type.

Multidimensional subshifts of finite type are not well understood. For example, it is not easy to describe their languages. Moreover, in addition to factors, there exist other types of dynamical transformations on multidimensional subshift: the sub-action of a d-dimensional

1998 ACM Subject Classification: G.2.m.
Key words and phrases: tiling, subshift, Turing machine with oracle, subdynamics.

tiling consists in taking the restriction of a tiling to a subgroup of $\mathbb{Z}^d$. Hochman showed that every d-dimensional subshift whose set of forbidden patterns is recursively enumerable can be obtained by sub-action and factor of a $d+2$-subshift of finite type [Hoc07].

This result suggests that a subshift can simulate another one, where the notion of simulation is given by operations on subshifts inspired by the dynamical theory. This involves different orders depending on the operations which are considered. In this paper, we present five types of operations: product, factor, finite type, sub-action and superposition. It is possible to formulate classic results with this formalism. Our main result (Theorem 4.2) establishes a correspondance between an order on subshifts based on dynamical transformations on them and an order on languages of forbidden patterns based on computability properties.

The paper is organized as follows: Section 1 is devoted to introduce the concepts of tiling and subshift. In Section 2, we present several operations on subshift which allow to define the notion of simulation of a subshift by another one. Then, in Section 3, we define an important tool to define runs of a Turing machine with a sofic subshift. This tool is used to proove our main result in the last Section.

1. Definitions

1.1. Generalized tilings

Let $\mathcal{A}$ be a finite alphabet and d be a positive integer. A *configuration* x is an element of $\mathcal{A}^{\mathbb{Z}^d}$. Let $\mathbb{S}$ be a finite subset of $\mathbb{Z}^d$. Denote $x_{\mathbb{S}}$ the *restriction* of x to $\mathbb{S}$. A *pattern* is an element $p \in \mathcal{A}^{\mathbb{S}}$ and $\mathbb{S}$ is the *support* of p, which is denoted by $\mathrm{supp}(p)$. For all $n \in \mathbb{N}$, we call $\mathbb{S}_n^d = [-n;n]^d$ the *elementary support* of size n. A pattern with support $\mathbb{S}_n^d$ is an *elementary pattern*. We denote by $\mathcal{E}_{\mathcal{A}}^d = \quad {}_{n\in\mathbb{N}} \mathcal{A}^{[-n;n]^d}$ the set of d-dimensional elementary patterns. A *d-dimensional language* $\mathcal{L}$ is a subset of $\mathcal{E}_{\mathcal{A}}^d$. A pattern p of support $\mathbb{S} \subset \mathbb{Z}^d$ *appears* in a configuration x if there exists $i \in \mathbb{Z}^d$ such that for all $j \in \mathbb{S}$, $p_j = x_{i+j}$, we note $p \sqsubset x$.

Definition 1.1. A *tile set* is a tuple $\tau = (\mathcal{A}, P)$ were P is a subset of $\mathcal{E}_{\mathcal{A}}^d$ called the *set of forbidden patterns.*

A *generalized tiling* by τ is a configuration x such that for all $p \in P$, p does not appear in x. We denote by $\mathbf{T}_\tau$ the set of generalized tilings by τ. If there is not ambiguity on the alphabet, we just denote it by $\mathbf{T}_P$.

Remark 1.2. If P is finite, it is equivalent to define a generalized tiling by allowed patterns or forbidden patterns, the latter being the usual definition of tiling.

1.2. Dynamical point of view : subshifts

One can define a toplogy on $\mathcal{A}^{\mathbb{Z}^d}$ by endowing $\mathcal{A}$ with the discrete topology, and considering the product topology on $\mathcal{A}^{\mathbb{Z}^d}$. For this topology, $\mathcal{A}^{\mathbb{Z}^d}$ is a compact metric space on which $\mathbb{Z}^d$ acts by translation via σ defined by:

$$\begin{array}{rccl} \sigma_{\mathcal{A}}^i : & \mathcal{A}^{\mathbb{Z}^d} & \longrightarrow & \mathcal{A}^{\mathbb{Z}^d} \\ & x & \longmapsto & \sigma_{\mathcal{A}}^i(x) \quad \text{such that } \sigma_{\mathcal{A}}^i(x)_u = x_{i+u} \ \forall u \in \mathbb{Z}^d. \end{array}$$

for all i in $\mathbb{Z}^d$. This action is called the shift.

Definition 1.3. A d-dimensional subshift on the alphabet $\mathcal{A}$ is a closed and σ-invariant subset of $\mathcal{A}^{\mathbb{Z}^d}$. We denote by $\mathcal{S}$ (resp. $\mathcal{S}_d$, $\mathcal{S}_{\leq d}$) the set of all subshifts (resp. d-dimensional subshifts, d'-dimensional subshifts with $d' \leq d$).

Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ be a subshift. Denote $\mathcal{L}_n(\mathbf{T}) \subseteq \mathcal{A}^{[-n;n]^d}$ the set of elementary patterns of size n wich appear in some element of $\mathbf{T}$, and $\mathcal{L}(\mathbf{T}) = {}_{n\in\mathbb{N}} \mathcal{L}_n(\mathbf{T})$ the *language* of $\mathbf{T}$ which is the set of elementary patterns which appears in some element of $\mathbf{T}$.

It is also usual to study subshift as dynamical system [LM95, Kit98], the next proposition shows the link between the two notions.

Proposition 1.4. *The set $\mathbf{T} \subset \mathcal{A}^{\mathbb{Z}^d}$ is a subshift if and only if $\mathbf{T} = \mathbf{T}_{\mathcal{L}(\mathbf{T})^c}$ where $\mathcal{L}(\mathbf{T})^c$ is the complement of $\mathcal{L}(\mathbf{T})$ in $\mathcal{E}^d_{\mathcal{A}}$.*

Definition 1.5. Let $\mathcal{A}$ be a finite alphabet and $\mathbf{T} \subset \mathcal{A}^{\mathbb{Z}^d}$ be a subshift.

The subshift $\mathcal{A}^{\mathbb{Z}^d}$ is the *full-shift* associated to $\mathcal{A}$. Denote $\mathcal{FS}$ the set of all full-shift.

If there exists a finite set $P \subseteq \mathcal{E}^d_{\mathcal{A}}$ such that $\mathbf{T} = \mathbf{T}_P$ then $\mathbf{T}$ is a *subshift of finite type.* Denote $\mathcal{SFT}$ the set of all subshift of finite type. Subshift of finite type correspond to the usual notion of tiling.

If there exists a recursively enumerable set $P \subseteq \mathcal{E}^d_{\mathcal{A}}$ such that $\mathbf{T} = \mathbf{T}_P$ then $\mathbf{T}$ is a *recursive enumerable subshift.* Denote $\mathcal{RE}$ the set of all recursive enumerable subshift.

2. Operations on tilings

2.1. Simulation of a tiling by another one

An *operation op* on subshifts transforms a subshift or a pair of subshifts into another one; it is a function $op : \mathcal{S} \to \mathcal{S}$ or $op : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$. We remark that a subshift $\mathbf{T}$ (resp. a pair of subshifts $(\mathbf{T}', \mathbf{T}''))$ and the image by an operation $op(\mathbf{T})$ (resp. $op(\mathbf{T}', \mathbf{T}''))$ do not necessary have the same alphabet or dimension. An operation can depend on a parameter.

Let Op be a set of operations on subshifts. Let $\mathcal{U} \subset \mathcal{S}$ be a set of subshifts. We define the *closure* of $\mathcal{U}$ under a set of operations Op, denoted by $\mathcal{C}l_{Op}(\mathcal{U})$, as the smallest set stable by Op which contains $\mathcal{U}$.

We say that a subshift $\mathbf{T}$ *simulates* a subshift $\mathbf{T}'$ by Op if $\mathbf{T}' \in \mathcal{C}l_{Op}(\mathbf{T})$. Thus there exists a finite sequence of operations chosen among Op, that transforms $\mathbf{T}$ into $\mathbf{T}'$. We note it by $\mathbf{T}' \leq_{Op} \mathbf{T}$. We remark that $\mathcal{C}l_{Op}(\mathbf{T}) = \{\mathbf{T}' \ : \ \mathbf{T}' \leq_{Op} \mathbf{T}\}$.

2.2. Local transformations

We describe three operations that modify locally the subshift.

• **Product** P**:**

Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ and $\mathbf{T}' \subseteq \mathcal{B}^{\mathbb{Z}^d}$ be two subshifts of the same dimension, define:

$$\phi_P(\mathbf{T}, \mathbf{T}') = \mathbf{T} \times \mathbf{T}' \subseteq (\mathcal{A} \times \mathcal{B})^{\mathbb{Z}^d}.$$

One has $\mathcal{C}l_P(\mathcal{FS}) = \mathcal{FS}$ and $\mathcal{C}l_P(\mathcal{SFT}) = \mathcal{SFT}$.

• **Finite type FT:**

These operations consist in adding a finite number of forbidden patterns to the initial subshift. Formally, let $\mathcal{A}$ be an alphabet, $P \subseteq \mathcal{E}^d_{\mathcal{A}}$ be a finite subset and let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ be a subshift. By proposition 1.4, there exists P' such that $\mathbf{T} = \mathbf{T}_{P'}$. Define:

$$\phi_{FT}(P, \mathbf{T}) = \mathbf{T}_{P \cup P'}.$$

If P and $\mathbf{T}$ have not the same alphabet or the same dimension, put $\phi_{FT}(P,\mathbf{T}) = \mathbf{T}$. We remark that $\phi_{FT}(P,\mathbf{T})$ could be empty if P prohibit too many patterns. By FT, one lists all operations on subshifts which are obtained by ϕ_{FT}.

By definition of subshift of finite type, one has $\mathcal{C}l_{FT}(\mathcal{FS}) = \mathcal{SFT}$.

• **Factor F:**

These operations allow to change the alphabet of a subshift by local modifications. Let $\mathcal{A}$ and $\mathcal{B}$ be two finite alphabets. A *morphism* $\pi : \mathcal{A}^{\mathbb{Z}^d} \to \mathcal{B}^{\mathbb{Z}^d}$ is a continuous function which commutes with the shift action (i.e. $\sigma^i \circ \pi = \pi \circ \sigma^i$ for all $i \in \mathbb{Z}^d$). In fact, such function can be defined locally [Hed69]: that is to say, there exist $\mathbb{U} \subset \mathbb{Z}^d$ finite, called *neighborhood*, and $\overline{\pi} : \mathcal{A}^{\mathbb{U}} \to \mathcal{B}$, called *local function*, such that $\pi(x)_i = \overline{\pi}(x_{i+\mathbb{U}})$ for all $i \in \mathbb{Z}^d$. Let $\mathbf{T}$ be a subshift, define:

$$\phi_F(\pi,\mathbf{T}) = \pi(\mathbf{T}).$$

If the domain of π and $\mathbf{T}$ do not have the same alphabet or the same dimension, put $\phi_F(\pi,\mathbf{T}) = \mathbf{T}$. By F, one lists all operations on subshifts which are obtained by ϕ_F.

One verifies that $\mathcal{C}l_F(\mathcal{SFT}) \neq \mathcal{SFT}$ (see Example **??** in the annex).

Definition 2.1. A sofic subshift is a factor of a subshift of finite type. Thus, the set of sofic subshifts is $\mathcal{S}ofic = \mathcal{C}l_F(\mathcal{SFT})$.

2.3. Transformation on the group of the action

We describe two operations that modify the group on which the subshift is defined, thus we change the dimension of the subshift.

• **Sub-action SA:**

These operations allow to take the restriction of a subshift of $\mathcal{A}^{\mathbb{Z}^d}$ according to a subgroup of $\mathbb{Z}^d$. Let $\mathbb{G}$ be a sub-group of $\mathbb{Z}^d$ generated by $u_1, u_2, \ldots, u_{d'}$ ($d' \leq d$). Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ be a subshift, define:

$$\phi_{SA}(\mathbb{G},\mathbf{T}) = \left\{ y \in \mathcal{A}^{\mathbb{Z}^{d'}} \ : \ \exists x \in \mathbf{T} \text{ such that } \forall i_1,\ldots,i_{d'} \in \mathbb{Z}^{d'}, y_{i_1,\ldots,i_{d'}} = x_{i_1u_1+\cdots+i_{d'}u_{d'}} \right\}.$$

It is easy to proof that $\phi_{SA}(\mathbb{G},\mathbf{T})$ is a subshift of $\mathcal{A}^{\mathbb{Z}^{d'}}$. If $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ and $\mathbb{G}$ is not a subgroup of $\mathbb{Z}^d$, put $\phi_{SA}(\mathbb{G},\mathbf{T}) = \mathbf{T}$. By SA, one lists all operations on subshifts which are obtained by ϕ_{SA}.

One verifies that $\mathcal{C}l_{SA}(\mathcal{SFT}) \neq \mathcal{SFT}$ and $\mathcal{C}l_{SA}(\mathcal{SFT}) \neq \mathcal{S}ofic$ (see respectively Example **??** and Example **??** in the annex).

Theorem 2.2. $\mathcal{C}l_{SA}(\mathcal{RE}) = \mathcal{RE}$.

• **Superposition SP:**

These operations increase the dimension of a subshift by a superposition of the initial subshift. Let $d, d' \in \mathbb{N}^*$. Let $\mathbb{G}$ and $\mathbb{G}'$ be two subgroups of $\mathbb{Z}^{d+d'}$ such that $\mathbb{G}$ is isomorphic to $\mathbb{Z}^d$ and $\mathbb{G} \oplus \mathbb{G}' = \mathbb{Z}^{d+d'}$. Let $\mathbf{T} \subseteq \mathcal{A}^{Z^d}$ be a subshift, define:

$$\phi_{SP}(\mathbb{G},\mathbb{G}',\mathbf{T}) = \left\{ x \in \mathcal{A}^{\mathbb{Z}^{d+d'}} \ : \ \forall i \in \mathbb{G}', x_{i+\mathbb{G}} \in \mathbf{T} \right\}.$$

If $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ and $\mathbb{G}$ is not isomorphic to $\mathbb{Z}^d$ or $\mathbb{G} \oplus \mathbb{G}' \neq \mathbb{Z}^{d+d'}$, put $\phi_{SP}(\mathbb{G},\mathbb{G}',\mathbf{T}) = \mathbf{T}$. By SP, one lists all operations on subshifts which are obtained by ϕ_{SP}.

It is easy to verify that $\mathcal{C}l_{SP}(\mathcal{SFT}) = \mathcal{SFT}$.

With this formalism, the result of M. Hochman [Hoc07] can be writen:

$$\mathcal{C}l_{F,SA}(\mathcal{SFT}) = \mathcal{RE}.$$

More precisely, he proves that $\mathcal{C}l_{F,SA}(\mathcal{SFT} \cap \mathcal{S}_{d+2}) \cap \mathcal{S}_{\leq d} = \mathcal{RE} \cap \mathcal{S}_{\leq d}$.

3. Simulation of Turing machines by subshifts

A Turing machine is a model of calculation defined by local rules. It seems natural to represent the runs of a machine by a 2-dimensional subshift: one dimension representing the tape and the other time evolution. But the main problem is that in general the Turing machine uses a finite part of the space-time diagram which is represented by the subshift. Robinson [**?**] proposes a self-similar structure to construct an aperiodic subshift of finite type of dimension 2. In fact, it is also possible to use a general construction with substitutions due to Mozes [**?**]. This construction allows to give to the machine finite spaces on which it calculates independently. The problem is that we cannot control the entry of the Turing machine in view to recognize a configuration of a subshift. To obtain this property, Hochman [Hoc07] uses similar tools to construct a sofic subshift of dimension 3 in order to to prove that $\mathcal{C}l_{F,SA}(\mathcal{SFT}) = \mathcal{RE}$. In this Section, we present a similar construction which is used to prove our main result in Section 4.

3.1. Substitution tilings

Let $\mathcal{A}$ be a finite alphabet. A *substitution* is a function $s : \mathcal{A} \to \mathcal{A}^{\mathbb{U}_k}$ where $\mathbb{U}_k = [1;k] \times [1;k]$. We naturally extend s to a function $s^n : \mathcal{A}^{\mathbb{U}_n} \to \mathcal{A}^{\mathbb{U}_{nk}}$ by identifying $\mathcal{A}^{\mathbb{U}_{nk}}$ with $(\mathcal{A}^{\mathbb{U}_k})^{\mathbb{U}_n}$. Starting from a letter placed in $(1,1) \in \mathbb{Z}^2$ and applying successively $s, s^k, \dots, s^{k^{n-1}}$ we obtain a sequence of patterns in $\mathcal{A}^{\mathbb{U}_{k^i}}$ for $i \in \{0, \dots, n\}$. Such patterns are called *s-patterns.*

Definition 3.1. The subshift $\mathbf{S}_s$ defined by the substitution s is

$$\mathbf{S}_s = \left\{ x \in \mathcal{A}^{\mathbb{Z}^2} \ : \ \text{every finite pattern of } x \text{ appears in a } s\text{-pattern} \right\}.$$

3.2. A framework for Turing machines

We now describe a family of substitutions s_n defined on the alphabet $\{\circ, \bullet\}$, which are used by M. Hochman [Hoc07] to prove $\mathcal{C}l_{F,SA}(\mathcal{SFT}) = \mathcal{RE}$. For every integer n the substitution s_n is given by :

$$\circ \longmapsto \begin{array}{cccc} \circ & \dots & \circ & \circ \\ \vdots & \iddots & \bullet & \circ \\ \circ & \iddots & \iddots & \vdots \\ \bullet & \circ & \dots & \circ \end{array} \quad \text{and} \quad \bullet \longmapsto \begin{array}{cccc} \circ & \dots & \circ & \bullet \\ \vdots & \iddots & \bullet & \circ \\ \circ & \iddots & \iddots & \vdots \\ \bullet & \circ & \dots & \circ \end{array}$$

where the patterns are of size $n \times n$. Let $\mathbf{S}_n$ be the tiling defined by substitution s_n.

These substitutions have good properties, in particular they are unique derivation substitutions and for this reason they verify [**?**]; one obtains:

Proposition 3.2. *For every integer n, there exists a SFT $\tilde{\mathbf{S}}_n$ and a letter to letter morphism π_n such that $\mathbf{S}_n = \pi_n(\tilde{\mathbf{S}}_n)$.*

Definition 3.3. If $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^2}$ is a subshift, we define $\mathbf{T}^{(\uparrow)}$ by :

$$\mathbf{T}^{(\uparrow)} = \left\{ x \in \mathcal{A}^{\mathbb{Z}^2} \ : \ \exists y \in \mathbf{T}, \forall (i,j) \in \mathbb{Z}^2, x_{(i,j)} = y_{(i,j-i)} \right\}.$$

Notice that if $\mathbf{T}$ is an SFT, then $\mathbf{T}^{(\uparrow)}$ is also an SFT (just shift the forbidden patterns of $\mathbf{T}$ to get those of $\mathbf{T}^{(\uparrow)}$).

We now work on the space $\mathbb{Z}^3 = \mathbb{Z}e_1 \oplus \mathbb{Z}e_2 \oplus \mathbb{Z}e_3$ and we construct the SFT $\mathbf{W}_2$, $\mathbf{W}_3$ and $\mathbf{W}_5 \subseteq \{\circ, \bullet\}^{\mathbb{Z}^3}$ defined by :

$$x \in \mathbf{W}_2 \Longleftrightarrow \begin{cases} \forall k \in \mathbb{Z}, x_{|\mathbb{Z}^2 \times \{k\}} \in \mathbf{S}_2^{(\uparrow)} \\ \forall u \in \mathbb{Z}^3, x_u = x_{u+e_3} \ (*) \end{cases} \qquad x \in \mathbf{W}_3 \Longleftrightarrow \begin{cases} \forall j \in \mathbb{Z}, x_{|\mathbb{Z} \times \{j\} \times \mathbb{Z}} \in \mathbf{S}_3^{(\uparrow)} \\ \forall u \in \mathbb{Z}^3, x_u = x_{u+e_2} \ (**) \end{cases}$$

$$x \in \mathbf{W}_5 \Longleftrightarrow \begin{cases} \forall k \in \mathbb{Z}, x_{|\mathbb{Z}^2 \times \{k\}} \in \mathbf{S}_5^{(\uparrow)} \\ \forall u \in \mathbb{Z}^3, x_u = x_{u+e_3} \ (***) \end{cases}$$

Let x be a configuration of the subshift $\mathbf{W}_2 \times \mathbf{W}_3 \times \mathbf{W}_5 \subseteq (\{\circ, \bullet\}^3)^{\mathbb{Z}^3}$. If we focus on the subshift $\mathbf{W}_3 \times \mathbf{W}_5$, we can see rectangles whose corners are defined by the letter $(\bullet, \bullet)$ of $\{\circ, \bullet\}^2$. These rectangles of size $5^n \times 3^m$ are spaces of calculation on which the Turing machine runs independently. On top of that the information brought by $\mathbf{W}_2$ gives the size of the entry pattern p on each rectangle : scanning the base of a rectangle from left to right, the entry word is located between the left corner and the first symbol $\bullet$ due to $\mathbf{W}_2$ that occurs. This results are resumed in proposition 3.4 and figures can be found in appendix **??**.

Proposition 3.4. *The product $\mathbf{W}_2 \times \mathbf{W}_3 \times \mathbf{W}_5$ is a partition of the space into rectangles, in which each plane $\{i\} \times \mathbb{Z}^2$ is paved by rectangles of same width and height. Moreover if there is a $5^m \times 3^p$-rectangle in $(i,j,k) \in \mathbb{Z}^3$ with entry of size 2^n, then there exists i' and i'' such that there exists a $5^{m+1} \times 3^p$-rectangle in (i',j,k) and a $5^m \times 3^{p+1}$-rectangle in (i'',j,k) both with entry of size 2^n.*

This result will be used in Section 4.2.2 to prove that, thanks to these arbitrary large rectangles, one can simulate a calculation with an arbitrary number of steps.

3.3. A 2-dimensional sofic subshift

We now explain how we can use the previously constructed framework to simulate a Turing machine by a subshift. First we recall the formal definition of a Turing machine.

Definition 3.5. Let $\mathcal{M} = (Q, \mathcal{A}, \Gamma, \sharp, q_0, \delta, Q_F)$ be a Turing machine, where :

- Q is a finite set of states; $q_0 \in Q$ is the initial state;
- $\mathcal{A}$ are Γ are two finite alphabets such that $\mathcal{A} \subsetneq \Gamma$;
- $\sharp \notin \Gamma$ is the blank symbol;
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, \cdot, \rightarrow\}$ is the transition function;
- $F \subset Q_F$ is the set of final states.

We can describe its behaviour with a set of 2-dimensional patterns. First dimension stands for the tape and second dimension for time evolution. For example the rule $\delta(q_1, x) = (q_2, y, \leftarrow)$ will be coded by :

(q_2, z)	y	z'
z	(q_1, x)	z'

Denote by $P_{\mathcal{M}}$ the set of forbidden patterns constructed according to the rules of $\mathcal{M}$. One can consider the subshift of finite type $\mathbf{T}_{P_{\mathcal{M}}}$ where each local pattern correspond to calculations of the machine $\mathcal{M}$. Then thanks to a product operation we superimpose these calculations on the framework, with the following finite conditions :

- condition **Init** : to copy out the entry word ;
- condition **Head** : the initial state q_0 appears on every rectangle down left corner and only here;
- condition **Stop** : when a side of a rectangle is reached by th head of the machine, the calculation stops and if necessary the tape contents is just copied out until the top of the rectangle;
- condition **Final** : when a final state is reached, the tape contents is just copied out for next steps of calculation until the top of the rectangle.

Define $\mathbf{T}_{\mathcal{M}}$ the subshift:

$$\mathbf{T}_{\mathcal{M}} = \phi_{FT}\left(\{\mathbf{Init}, \mathbf{Head}, \mathbf{Stop}, \mathbf{Final}\}, \mathcal{A}^{\mathbb{Z}^3} \times (\mathbf{W}_2 \times \mathbf{W}_3 \times \mathbf{W}_5) \times \phi_{SP}(\mathbb{Z}e_2 \oplus \mathbb{Z}e_3, \mathbb{Z}e_1, \mathbf{T}_{P_{\mathcal{M}}})\right).$$

By stability of the class of subshift of finite type by SP, $\mathbf{T}_{\mathcal{M}}$ is a subshift of finite type up to a letter to letter morphism; thus $\mathbf{T}_{\mathcal{M}} \in \mathcal{S}ofic$. For all $i \in \mathbb{Z}$, in the plane $\{i\} \times \mathbb{Z}^2$, it is possible to find rectanges of size $5^m \times 3^p$ arbitrary large and an entry of size 2^n also arbitray large. On each rectangle, thanks to the conditions $P_{\mathcal{M}}$, we can observe the evolution of the Turing machine $\mathcal{M}$ - see the figure in Appendix **??**.

Remark 3.6. The construction described here only works for usual Turing machine. In Section 4.2.2 we explain how to add finite conditions on the subshift $\mathbf{T}_{\mathcal{M}}$ if $\mathcal{M}$ is a Turing machine with oracle.

4. Study of the semi-order $\leq_{P,F,FT,SA,SP}$

In this section we focuse on the five operations described previously. Our aim is to study the semi-order $\leq_{P,F,FT,SA,SP}$.

4.1. A semi-order on languages

A Turing machine with *semi-oracle* is a usual machine with on top of that a special state $q_?$ and an oracle tape. The behaviour of a Turing machine with semi-oracle $\mathcal{L}$, where $\mathcal{L}$ is a language, is the following : the machine reads an entry pattern p and writes a pattern on the oracle tape, until the state $q_?$ is reached. If the pattern written on the oracle tape is in $\mathcal{L}$ then the machine stops, else it keeps on calculating.

We define a semi-order on languages :

$$\mathcal{L} \preceq \mathcal{L}' \Longleftrightarrow \exists \mathcal{M}^{\mathcal{L}'} \text{ a Turing machine with semi-oracle } \mathcal{L}' \text{ such that } dom(\mathcal{M}^{\mathcal{L}'}) = \mathcal{L},$$

where $dom(\mathcal{M})$ is the domain of the machine $\mathcal{M}$ that is to say the set of entry words on which $\mathcal{M}$ stops. We refer to [**?**] for definitions and properties of similar semi-orders on languages based on computability.

Proposition 4.1. *$\preceq$ is a semi-order.*

Consider the equivalence relation $\mathcal{L} \approx \mathcal{L}'$ if and only if $\mathcal{L} \preceq \mathcal{L}'$ and $\mathcal{L}' \preceq \mathcal{L}$. This equivalence relation defines classes of languages, and we can compare them within the semi-order. For instance, the class of recursively enumerable languages is the smallest for this semi-order. We have $\emptyset \approx \mathcal{L}$ for every recursively enumerable language $\mathcal{L}$.

4.2. Closure theorem:

The semi-order on languages defined by semi-oracle Turing machines corresponds to a semi-order on subshifts:

Theorem 4.2. *Let* $\mathbf{T}$ *be a subshift, one has:*

$$\mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T}) = \{\mathbf{T}_{\mathcal{L}} : \mathcal{L} \preceq \mathcal{L}(\mathbf{T})^c\}.$$

Or equivalently, if $\mathbf{T}'$ *and* $\mathbf{T}''$ *are two subshifts of dimension* d' *and* d''*, one has:*

$$\mathbf{T}' \leq_{P,F,FT,SA,SP} \mathbf{T}'' \Longleftrightarrow \mathcal{L}(\mathbf{T}')^c \preceq \mathcal{L}(\mathbf{T}'')^c.$$

4.2.1. *Direct inclusion.* Put $\mathcal{L} = \mathcal{L}(\mathbf{T})^c$. To show $\mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T}) \subseteq \{\mathbf{T}_{\mathcal{L}'} : \mathcal{L}' \preceq \mathcal{L}\}$, it is sufficent to show the stability of $\{\mathbf{T}_{\mathcal{L}'} : \mathcal{L}' \preceq \mathcal{L}\}$ by all the operations. Let $\mathcal{L}_1 \subseteq \mathcal{E}_{\mathcal{A}_1}^{d_1}$ and $\mathcal{L}_2 \subseteq \mathcal{E}_{\mathcal{A}_2}^{d_2}$ be two languages such that $\mathcal{L}_i \preceq \mathcal{L}$ for $i \in \{1,2\}$. Thus, for $i \in \{1,2\}$, there exists Turing machine $\mathcal{M}_i$ with semi-oracle $\mathcal{L}$ whose domain is exactly $\mathcal{L}_i$.

• **Stability under product:** Let $\mathbf{T}' = \phi_P(\mathbf{T}_1, \mathbf{T}_2)$, so $\mathbf{T}' = \mathbf{T}_{\mathcal{L}'}$ with $\mathcal{L}' = \mathcal{L}_1 \times \mathcal{E}_{\mathcal{A}_2}^{d_2} \cup \mathcal{E}_{\mathcal{A}_1}^{d_1} \times \mathcal{L}_2$. The language $\mathcal{L}'$ could be the domain of a Turing machine $\mathcal{M}'$ with semi-oracle $\mathcal{L}$. It suffies to simulate the two Turing machines $\mathcal{M}_1$ and $\mathcal{M}_2$ (each machine runs during one step successively) on each coordinates of a pattern of $\mathcal{L}'$. Thus $\mathcal{L}' \preceq \mathcal{L}$.

• **Stability under finite type:** Let $\mathbf{T}' = \phi_{FT}(P, \mathbf{T}_{\mathcal{L}_1})$. Since P is finite, one has $\mathcal{L}_1 \cup P \preceq \mathcal{L}_1 \preceq \mathcal{L}$ and $\mathbf{T}' = \mathbf{T}_{\mathcal{L}_1 \cup P}$.

• **Stability under factor map:** Let $\mathbf{T}' = \phi(\pi, \mathbf{T}_{\mathcal{L}_1})$ where $\pi : \mathcal{A}_1^{\mathbb{Z}^{d_1}} \to \mathcal{B}^{\mathbb{Z}^{d_1}}$ is a morphism of neighborhood $\mathbb{S}_n^{d_1}$ and local function $\overline{\pi}$. One has $\mathbf{T}' = \mathbf{T}_{\mathcal{L}'}$ where $\mathcal{L}' = (\overline{\pi}(\mathcal{L}_1^c))^c$. Moreover, one has $\mathcal{L}' \preceq \mathcal{L}_1$. Indeed, if $p \in \mathcal{E}_{\mathcal{B}}^{d_1}$, we simulate the machine $\mathcal{M}_1$ on all pattern $p' \in \mathcal{A}^{supp(p)+\mathbb{S}_n^{d_1}}$ such that $\overline{\pi}(p') = p$, running successively one step for each pattern.

• **Stability under sub-action:** Let $\mathbf{T}' = \phi_{SA}(\mathbb{G}, \mathbf{T}_{\mathcal{L}_1}) \subseteq \mathcal{A}_1^{\mathbb{Z}^{d'}}$ where $\mathbb{G}$ is a subgroup of $\mathbb{Z}^{d_1}$ of dimension $d' \leq d_1$. We consider the language $\mathcal{L}' \subseteq \mathcal{E}_{\mathcal{A}_1}^{d_1}$ which is the domain of the Turing machine $\mathcal{M}'$: on a pattern $p \in \mathcal{E}_{\mathcal{A}_1}^{d'}$ of support $\mathbb{U}$, a Turing machine $\mathcal{M}'$ simulates successively $\mathcal{M}_1$ on every entry word of support $[-n;n]^{d_1}$ which completes p in $\mathcal{E}_{\mathcal{A}_1}^{d_1}$ where $[-n;n]^{d_1}$ is the minimal support which contains $\mathbb{U}$ embedded in $\mathbb{G}$. Thus $\mathcal{L}' \preceq \mathcal{L}_1$, moreover $\mathbf{T}' = \mathbf{T}_{\mathcal{L}'}$. This is exactly the same principle as in the proof of Theorem 2.2.

• **Stability under superposition:** Let $\mathbf{T}' = \phi_{SP}(\mathbb{G}, \mathbb{G}', \mathbf{T}_{\mathcal{L}_1})$ where $\mathbb{G}$ is isomorph to $\mathbb{Z}^{d_1}$ and $\mathbb{G} \oplus \mathbb{G}' = \mathbb{Z}^{d_1+d}$. Let $\mathcal{L}' \subseteq \mathcal{E}_{\mathcal{A}_1}^{d_1+d}$ be the language where each pattern p is the superposition of patterns $p_1, \dots, p_d \in \mathcal{E}_{\mathcal{A}_1}^{d_1}$ and there exists $i \in \{1, \dots, d\}$ such that $p_i \in \mathcal{L}_1$. Thus $\mathcal{L}' \preceq \mathcal{L}_1$ and $\mathbf{T}' = \mathbf{T}_{\mathcal{L}'}$.

4.2.2. *Reciprocal inclusion.* Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$be a subshift; define $\mathcal{L} = \mathcal{L}(\mathbf{T})^c \subseteq \mathcal{E}^d_{\mathcal{A}}$. Let $\mathcal{L}' \subseteq \mathcal{E}^{d'}_{\mathcal{B}}$ be a language such that $\mathcal{L}' \preceq \mathcal{L}$. We want to prove that $\mathbf{T}_{\mathcal{L}'} \in \mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T})$.

Here, we assume that $\mathcal{L}$ and $\mathcal{L}'$ are one-dimensional languages, but the proof can be adapted to the general case. We explain how to construct the subshift $\mathbf{T}_{\mathcal{L}'}$ thanks to operations P, F, FT, SA and SP applied on $\mathbf{T} = \mathbf{T}_{\mathcal{L}}$.

Since $\mathcal{L}' \preceq \mathcal{L}$ there exists a Turing machine $\mathcal{M}$ with semi-oracle $\mathcal{L}$ such that $dom(\mathcal{M}) = \mathcal{L}'$. We transform this Turing machine so that it only take in entrance patterns of support $[0, 2^{n-1}]$ (because checked patterns are given by $\mathbf{W}_2$) and at the moment when the state $q_?$ is reached, the word written on the oracle tape is copied out in the alphabet $\widetilde{\mathcal{A}}$, which is simply a copy of $\mathcal{A}$, then again copied out in the alphabet $\mathcal{A}$ once the oracle has given its answer.

We first list auxiliary subshifts that we need to construct $\mathbf{T}_{\mathcal{L}'}$:

- the original subshift $\mathbf{T}_{\mathcal{L}}$ written in the copy of $\mathcal{A}$: $\widetilde{\mathbf{T}}_{\mathcal{L}} \subseteq \widetilde{\mathcal{A}}^{\mathbb{Z}}$ will simulate the oracle;
- turing machine $\mathcal{M}$ is coded by a subshift of finite type $\mathbf{T}_{\mathcal{M}} \subseteq \mathcal{O}^{\mathbb{Z}^2}$, where $\mathcal{O}$ is an alphabet that contains at least $\mathcal{A}$, $\widetilde{\mathcal{A}}$ and $\mathcal{B}$;
- the framework for this Turing machine will be given by $\mathbf{W}_2$, $\mathbf{W}_3$ and $\mathbf{W}_5$ defined in Section 3; they are defined on the alphabet $\{\bullet, \circ\}$ and are subshifts of finite type up to a letter to letter morphism.

Construction of $\mathbf{T}_{\mathcal{L}'}$. The principle is to construct $\mathbf{\Sigma} \in \mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T}_{\mathcal{L}})$ a 4-dimensional subshift on the alphabet $\mathcal{C} = \mathcal{A} \times \widetilde{\mathcal{A}} \times \mathcal{B} \times \{\bullet; \circ\}^3 \times \mathcal{O}$. Denote (e_1, e_2, e_3, e_4) the canonical basis of $\mathbb{Z}^4$. We need these four dimensions for different reasons :

- the subshift $\mathbf{T}_{\mathcal{L}'}$ will appear on $\mathbb{Z}e_1$;
- thanks to $\mathbb{Z}e_1 \oplus \mathbb{Z}e_2 \oplus \mathbb{Z}e_3$, we construct a framework for $\mathcal{M}$, so that every rectangle of this framework is in a plane $\{i\} \times \mathbb{Z} \times \mathbb{Z} \times \{k\}$ where $i, k \in \mathbb{Z}$;
- on $\mathbb{Z}e_4$ we have the oracle simulated by $\widetilde{\mathbf{T}}_{\mathcal{L}}$.

Step 1 : First notice that changing $\mathbf{T}_{\mathcal{L}}$ into $\widetilde{\mathbf{T}}_{\mathcal{L}}$ only require a letter to letter morphism. Then we construct $\tilde{\mathbf{W}} = \phi_{SP}(\mathbb{Z}e_4, \mathbb{Z}e_1 \oplus \mathbb{Z}e_2 \oplus \mathbb{Z}e_3, \widetilde{\mathbf{T}}_{\mathcal{L}})$ to place $\widetilde{\mathbf{T}}_{\mathcal{L}}$ in a 4-dimensional subshift. We finally add through a product operation P all letters from $\mathcal{C}$: $\mathbf{W} = \tilde{\mathbf{W}} \times (\mathcal{A} \times \mathcal{B} \times \{\bullet; \circ\}^3 \times \mathcal{O})^{\mathbb{Z}^4}$ so that $\mathbf{W} \in \mathcal{C}l_{P,F,SP}(\mathbf{T}_{\mathcal{L}}) \cap \mathcal{C}^{\mathbb{Z}^4}$.

Step 2 : We want $\mathbf{T}_{\mathcal{L}'}$ to appear on $\mathbb{Z}e_1$. Simulations of the Turing machine $\mathcal{M}$ will take in entrance a word written on $\mathbb{Z}e_2$. So we need to copy out $\mathbb{Z}e_1$ on $\mathbb{Z}e_2$ so that these simulations apply to what will be the subshift $X_{\mathcal{L}'}$. We get to it with the finite condition :

$$\forall x \in \mathcal{C}^{\mathbb{Z}^4}, \forall u \in \mathbb{Z}^4, x_u = x_{u+e_1-e_2}.$$

We also want to keep accessible all along the simulation the entry word of every rectangle of the framework. To do that we add the finite condition :

$$\forall x \in \mathcal{C}^{\mathbb{Z}^4}, \forall u \in \mathbb{Z}^4, x_u = x_{u+e_3}.$$

We thus obtain a subshift $\mathbf{W}' \in \mathcal{C}l_{P,F,SP,FT}(\mathbf{T}_{\mathcal{L}})$.

Step 3 : Then we add to $\mathbf{W}'$ a framework for the Turing machine. We construct $W_{\text{rect}} \subseteq \{\bullet,\circ\}^{\mathbb{Z}^3}$ an auxiliary subshift of finite type up to a letter to letter morphism, containing well-chosen rectangles. Denote F_i the finite type condition that ensures $\forall x \in \{\bullet,\circ\}^{\mathbb{Z}^3}, \forall u \in \mathbb{Z}^3, x_u = x_{u+e_i}$. As in Section 3, we define:

- $\mathbf{W}_2 = \phi_{FT}(F_3, \phi_{SP}(\mathbb{Z}e_1 \oplus \mathbb{Z}e_2, \mathbb{Z}e_3 \oplus \mathbb{Z}e_4, \mathbf{S}_2^{(\uparrow)}))$;
- $\mathbf{W}_5 = \phi_{FT}(F_3, \phi_{SP}(\mathbb{Z}e_1 \oplus \mathbb{Z}e_2, \mathbb{Z}e_3 \oplus \mathbb{Z}e_4, \mathbf{S}_5^{(\uparrow)}))$;
- $\mathbf{W}_3 = \phi_{FT}(F_2, \phi_{SP}(\mathbb{Z}e_1 \oplus \mathbb{Z}e_3, \mathbb{Z}e_2 \oplus \mathbb{Z}e_4, \mathbf{S}_3^{(\uparrow)}))$.

The rectangles are obtained in $\tilde{\mathbf{W}}_{\text{rect}} = \mathbf{W}_2 \times \mathbf{W}_5 \times \mathbf{W}_3$. Each rectangle of lenght 5^m given by $\mathbf{W}_5$ has on top of that the length of its entrance 2^n given by $\mathbf{W}_2$. Thus we can simulate the Turing machine on words of length 2^n, on a tape of length 5^m and simulations are bounded by 3^p steps of calculation. Up to a letter to letter morphism, $\tilde{\mathbf{W}}_{\text{rect}}$ is a subshift of finite type, so there exists a finite set of patterns F_{rect} and a morphism π_{rect} such that $\tilde{\mathbf{W}}_{\text{rect}} = \pi_{\text{rect}}(\mathbf{T}_{F_{\text{rect}}})$. We add this framework to $\mathbf{W}'$ via $\mathbf{W}_{\text{rect}} = \pi_{\text{rect}}(\phi_{FT}(F_{\text{rect}}, \mathbf{W}'))$ so that we have $\mathbf{W}_{\text{rect}} \in \mathcal{C}l_{P,F,SP,FT}(\mathbf{T}_{\mathcal{L}})$.

Step 4 : We add the behaviour of $\mathcal{M}$ in rectangles of $\mathbf{W}_{\text{rect}}$ but for the moment we do not take into consideration calls for oracle. As in Section 3, we consider the finite conditions $P_{\mathcal{M}}$ given by the rule of $\mathcal{M}$ and the conditions $P_{\text{calc}} = \{\mathbf{Init}, \mathbf{Head}, \mathbf{Stop}, \mathbf{Final}\}$ which control the interaction of the head of $\mathcal{M}$ with the rectangles. For the moment every time the machine calls the oracle it keeps on calculating. Thus $\mathbf{W}_{\mathcal{M}} = \phi_{FT}(P_{\mathcal{M}} \cup P_{\text{calc}}, \mathbf{W}_{\text{rect}}) \in \mathcal{C}l_{P,F,FT,SP}(\mathbf{T}_{\mathcal{L}})$.

Step 5 : To simulate the oracle, we add finite type conditions to ensure that during a calculation, when the machine calls for the oracle in $(i,j,k,l) \in \mathbb{Z}^4$, the pattern $p \in \tilde{\mathcal{A}}^n$ on which the oracle is called coincides with the pattern in $\mathbb{Z}e_4$ between (i,j,k,l) and $(i,j,k,l+n)$. These new allowed patterns look like :

$$\uparrow_{e_4}\ \underset{\rightarrow_{e_2}}{\begin{array}{|c|c|}\hline \tilde{a} & . \\ \hline \tilde{b} & \tilde{a} \\ \hline \end{array}}\ ,\quad \begin{array}{|c|c|}\hline \tilde{a} & . \\ \hline (q_?, \tilde{b}) & \tilde{a} \\ \hline \end{array}$$

However, these conditions are only valid in the interior of a rectangle. We denote these finite type conditions by F_{oracle}. Then we have $\mathbf{W}_{\mathcal{M}_{\text{oracle}}} = \phi_{FT}(F_{\text{oracle}}, W_{\mathcal{M}}) \in \mathcal{C}l_{P,F,SP,FT}(\mathbf{T}_{\mathcal{L}})$.

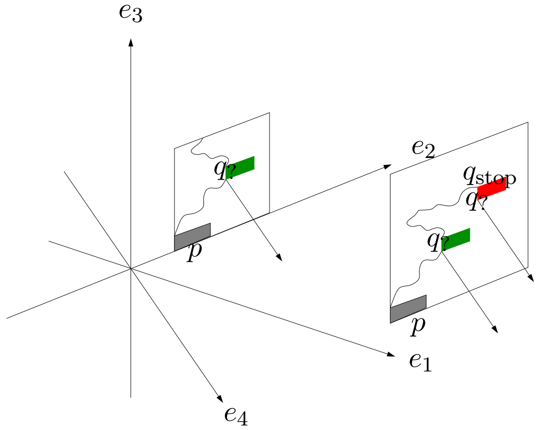
Step 6 : In order to avoid dependence problems between differents calculations, each configuration of $\mathbf{T}_{\mathcal{L}}$ that appears on $\mathbb{Z}^4$ is used for the same calculation, thanks to the finite type condition :

$$\forall x \in \mathcal{C}^{\mathbb{Z}^4}, \forall u \in \mathbb{Z}^4, x_u = x_{u+e_1+e_4}.$$

Finally we consider the final state q_{stop} as a forbidden pattern and we denote by $\mathbf{\Sigma}$ this subshift. We have $\mathbf{\Sigma} \in \mathcal{C}l_{P,F,SP,FT}(\mathbf{T}_{\mathcal{L}})$.

We simulate the running of the Turing machine $\mathcal{M}$ on a pattern $p \in \mathcal{E}^1_{\mathcal{B}}$ of length 2^n. As soon as $\mathcal{M}$ calls for the oracle, we compare the word on which the orcale is called and the word on $\mathbb{Z}e_4$. If the two words coincide then $\mathcal{M}$ keeps on calculating, else it come to the final state q_{stop}. If the machine cannot terminate its calculation within the time given by the rectangle, Proposition 3.4 ensures that we can find a larger rectangle in which the machine will calculate on the same entry word.

The following drawing resumes the behaviour of the machine $\mathcal{M}$ on the framework :



Proof that this construction works. We now prove that $\phi_{SA}(\mathbb{Z}e_1, \mathbf{\Sigma})$, the projection of $\mathbf{\Sigma}$ on $\mathbb{Z}e_1$ is $\mathbf{T}_{\mathcal{L}'}$, up to a morphism that just consists in keeping information about $\mathcal{B}$.

Proof of $\phi_{SA}(\mathbb{Z}e_1, \mathbf{\Sigma}) \subseteq \mathbf{T}_{\mathcal{L}'}$: Let $y \in \mathbf{\Sigma}$, we prove that $x = y_{|\mathbb{Z}e_1} \in \mathbf{T}_{\mathcal{L}'}$. It is sufficient to prove that every pattern in x is not in $\mathcal{L}'$. Let p be a pattern in x then it is a sub-pattern of a certain $p' \sqsubset x$ where p' is chosen such that it is of length 2^n. By construction of $\mathbf{W}_{\text{rect}}$ there exists $t, s \in \mathbb{N}$ arbitrarry large such that there exists a rectangle of size $5^s \times 3^t$ with the entry word p'. Since $y \in \mathbf{\Sigma}$, in every rectangle the calculation of the machine $\mathcal{M}$ on the word p' does not reach the final state q_{stop}. Since these rectangles are arbitrarily large, we can conclude that the machine $\mathcal{M}$ never reaches q_{stop}. It means that $p' \notin \mathcal{L}'$, thus $p \notin \mathcal{L}'$.

Proof of $\mathbf{T}_{\mathcal{L}'} \subseteq \phi_{SA}(\mathbb{Z}e_1, \mathbf{\Sigma})$: Let $x \in \mathbf{T}_{\mathcal{L}'}$, we construct $y \in \mathcal{C}^{\mathbb{Z}^4}$ such that $y \in \mathbf{\Sigma}$ and $y_{|\mathbb{Z}e_1} = x$. To insure that $y \in \mathbf{\Sigma}$ we just need to check that for all $(i,j,k) \in \mathbb{Z}^3$, we can impose that $y_{|\{i\}\times\{j\}\times\{k\}\times\mathbb{Z}} \in \mathbf{T}_{\mathcal{L}}$ while the calculations of $\mathcal{M}$ in the rectangles containing any (i,j,k,l) do not reach the state q_{stop}.

Let's now focus on a specific rectangle of the framework, on which the machine $\mathcal{M}$ calculates on a pattern p of size 2^n that appears in x. Since p appears in x, $p \notin \mathcal{L}'$ so the machine $\mathcal{M}$ loops on the entry p. It means that every time the calculation of $\mathcal{M}$ on p calls for the oracle on a pattern p', p' is not in $\mathcal{L}$. Since $\mathcal{L} = \mathcal{L}(\mathbf{T})^c$, for all pattern p' on which the oracle is called, there exists a configuration $z \in \mathbf{T}_{\mathcal{L}}$ such that $z_{|[0;|m'|-1]} = p'$. Thus we complete y on the following way :

- if in $(i,j,k) \in \mathbb{Z}^3$ the calculation of $\mathcal{M}$ calls for the oracle on a pattern p', then $y_{|\{i\}\times\{j\}\times\{k\}\times\mathbb{Z}} = z$ previously constructed;
- if the oracle is not called, we complete y with any $y_{|\{i\}\times\{j\}\times\{k\}\times\mathbb{Z}} \in \mathbf{T}_{\mathcal{L}}$.

This makes sure that y is in the subshift $\mathbf{\Sigma}$, so $x \in \phi_{SA}(\mathbb{Z}e_1, \mathbf{\Sigma})$.

The proof of Theorem is completed. ■

An application of Theorem 4.2: There does not exist an "universal" subshift $\mathbf{T}$ which could simulate every element of $\mathcal{S}$. Indeed, consider $\mathcal{L} = \mathcal{L}(\mathbf{T})^c$, one has $\mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T}) = \{\mathbf{T}_{\mathcal{L}'} : \mathcal{L}' \preceq \mathcal{L}\}$. But there exists $\mathcal{L}''$ strictly superior to $\mathcal{L}$ (see [**?**]). Moreover, one can choose $\mathcal{L}''$ such that for all patterns $p \in \mathcal{L}'' \subseteq \mathcal{E}^d_{\mathcal{A}}$, then for all $p' \in \mathcal{E}^d_{\mathcal{A}}$ such that $p \sqsubset p'$, one has $p' \in \mathcal{L}''$. Thus $\mathcal{L}(\mathbf{T}_{\mathcal{L}''})^c = \mathcal{L}''$. One deduces that $\mathbf{T}_{\mathcal{L}''} \notin \mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T})$.

Conclusion

In this article we generalize the notion of tilings considering any set of forbidden patterns. We present operations on sets of tilings, called subshifts, inspired by the dynamical theory. We obtain different notions of simulation, depending on the set of operations which are considered. These notions involve different semi-orders on subshifts and in this article we focus on the semi-order which consider all the transformations presented. This semi-order is quite well understood since we establish a correspondance with a semi-order on languages of forbidden patterns based on computability properties. The following points are still open questions :

- In our construction, considering two subshifts $\mathbf{T}_1$ and $\mathbf{T}_2$ respectively of dimension d_1 and d_2 such that $\mathcal{L}(\mathbf{T}_2)^c \preceq \mathcal{L}(\mathbf{T}_1)^c$, we need $\mathbf{\Sigma} \in \mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T}_1)$ of dimension $d_1 + d_2 + 2$ to simulate $\mathbf{T}_2$. It is possible to decrease the dimension of $\mathbf{\Sigma}$?
- For which class $\mathcal{U} \subseteq \mathcal{S}$ there exists a subshift $\mathbf{T}$ such that $\mathcal{C}l_{P,F,SA,SP,FT}(\mathbf{T}) = \mathcal{U}$?

We can also consider other semi-orders involved by other set of operations and look for general tools to study them. In fact, some of these semi-order have already been studied. For example, the set of space-time diagrams of a cellular automaton can be viewed as a subshift, and the orders presented in [**?**, **?**, **?**] could be formalized with the tools introduced in Section 2.

References

[BDJ08] Alexis Ballier, Bruno Durand, and Emmanuel Jeandel. Structural Aspects of Tilings. In *Proceedings of the 25th Symposium on Theoretical Aspects of Computer Science : STACS 2008*, 2008.

[Bea93] M.P. Beal. *Codage Symbolique*. Masson, 1993.

[Ber66] R. Berger. *The Undecidability of the Domino Problem*. American Mathematical Society, 1966.

[Han74] William Hanf. Nonrecursive tilings of the plane. i. *The Journal of Symbolic Logic*, 39(2):283–285, 1974.

[Hed69] GA Hedlund. Endomorphisms and automorphisms of the shift dynamical system. *Theory of Computing Systems*, 3(4):320–375, 1969.

[Hoc07] M. Hochman. On the Dynamics and Recursive Properties of Multidimensional Symbolic Systems. 2007.

[Kit98] B. Kitchens. *Symbolic dynamics*. Springer New York, 1998.

[LM95] D. Lind and B. Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1995.

[Moz89] S. Mozes. Tilings, substitution systems and dynamical systems generated by them. *Journal d'analyse mathématique(Jerusalem)*, 53:139–186, 1989.

[MR99] J. Mazoyer and I. Rapaport. Inducing an order on cellular automata by a grouping operation. *Discrete Applied Mathematics*, 91(1-3):177–196, 1999.

[Oll03] N. Ollinger. The intrinsic universality problem of one-dimensional cellular automata. *Symposium on Theoretical Aspects of Computer Science (STACS'2003)*, LNCS:632–641, 2003.

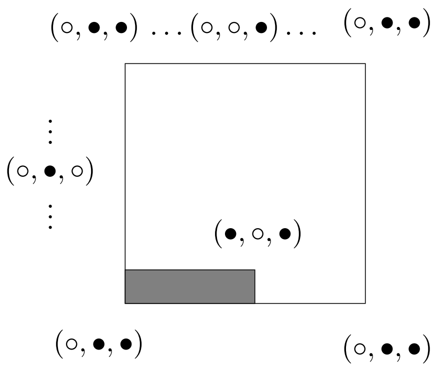
[RJ87] H. Rogers Jr. *Theory of recursive functions and effective computability*. MIT Press Cambridge, MA, USA, 1987.

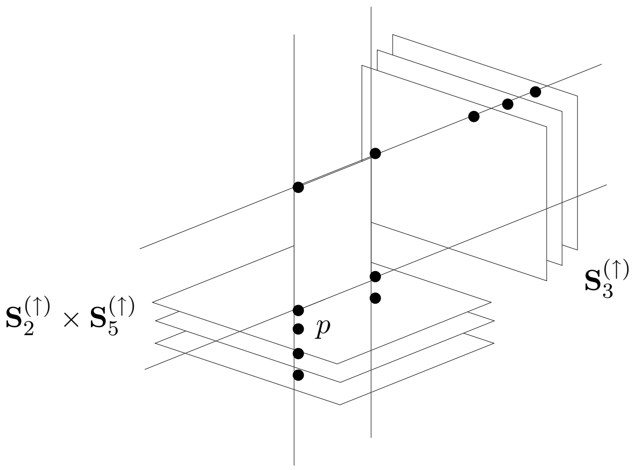
[Rob71] R.M. Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones Mathematicae*, 12(3):177–209, 1971.

[The05] G. Theyssier. *Automates cellulaires: un modèle de complexités*. PhD thesis, École Normale Supérieure de Lyon, 2005.

Appendix A. Figures

A.1. Construction of rectangles



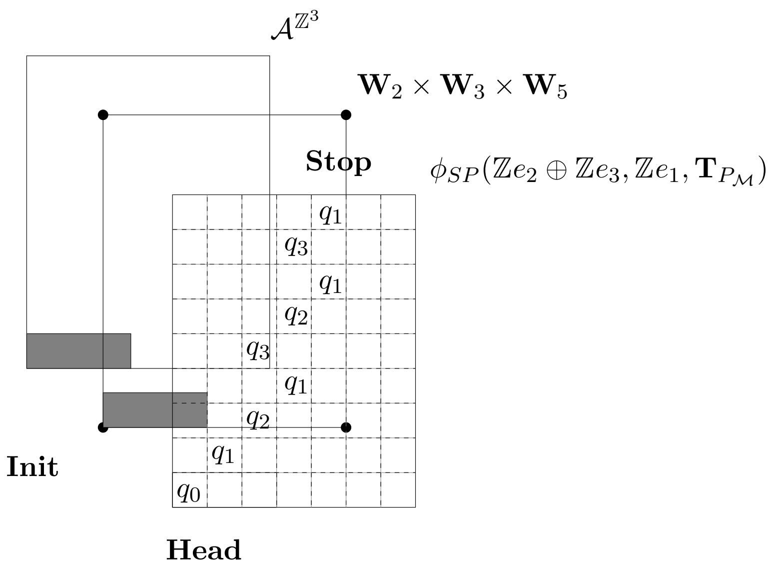


A.2. Simulation of a Turing machine

We present the construction to simulate a Turing machine thanks to the product subshift

$$\mathbf{T}_{\mathcal{M}} = \phi_{FT}\left(\{\mathbf{Init}, \mathbf{Head}, \mathbf{Stop}, \mathbf{Final}\}, \mathcal{A}^{\mathbb{Z}^3} \times (\mathbf{W}_2 \times \mathbf{W}_3 \times \mathbf{W}_5) \times \phi_{SP}(\mathbb{Z}e_2 \oplus \mathbb{Z}e_3, \mathbb{Z}e_1, \mathbf{T}_{P_{\mathcal{M}}})\right)$$

. The first part of this product $\mathcal{A}^{\mathbb{Z}^3}$ contains the entry word, which is copied out in $\mathbf{W}_2 \times \mathbf{W}_3 \times \mathbf{W}_5$ and in $\phi_{SP}(\mathbb{Z}e_2 \oplus \mathbb{Z}e_3, \mathbb{Z}e_1, \mathbf{T}_{P_{\mathcal{M}}})$ thanks to condition **Init**. The second part of the product $\mathbf{W}_2 \times \mathbf{W}_3 \times \mathbf{W}_5$ contains the framework which gives rectangles of calculation to the third subshift $\phi_{SP}(\mathbb{Z}e_2 \oplus \mathbb{Z}e_3, \mathbb{Z}e_1, \mathbf{T}_{P_{\mathcal{M}}})$. Finite conditions **Head**, **Stop** and **Final** ensure that the Turing machine runs independently on different rectangles.



Appendix B. Some counter-examples

B.1. $\mathcal{C}l_F(\mathcal{SFT}) \neq \mathcal{SFT}$

Consider the alphabet $\{0,1,2\}^{\mathbb{Z}}$ and define $\mathbf{T} = \mathbf{T}_{\{00,11,02,21\}}$. The factor π such that $\pi(0) = \pi(1) = 0$ and $\pi(2) = 2$ transforms $\mathbf{T}$ into a subshift :

$$\pi(\mathbf{T}) = \{x \in \{0,2\}^{\mathbb{Z}} \ : \ \text{blocks of consecutive 0 are of even length } \}$$

which is not a subshift of finite type, since one need to exclude arbitrary large blocks of consecutive 0 of odd lengths to describe it.

B.2. $\mathcal{C}l_{SA}(\mathcal{SFT}) \neq \mathcal{SFT}$

We construct a subshift of finite type $\mathbf{T} \subset \{0,1,2\}^{\mathbb{Z}^2}$ such that the sub-action of $\mathbf{T}$ on the sub-group $\Delta = \{(x,y) \in \mathbb{Z}^2 : y = x\} \subseteq \mathbb{Z}^2$ is not of finite type. In this example we want the subshift that appears on the staight line Δ to be

$$\{x \in \{0,1,2\}^{\mathbb{Z}} \ : \ \text{blocks of consecutive 0 are of even length }\}.$$

Define $\overline{F}$ a set of allowed patterns (. symbole may be 1 or 2 but not 0, blank symbole may be 0,1 or 2) :

		2	0
	1	0	1
2	0	2	
0	1		

;

		.	.
	2	0	.
1	0	1	
0	2		

;

		1	0
	2	0	2
.	0	1	
.	.		

;

		.	
	.	.	.
2	0	.	
0	1		

;

		.	
	.		.
.	.	.	
0	.		

;

		2	0
	.	0	1
.	.	.	
	.		

;

		.	0
	.	.	.
.		.	
	.		

The alternation of 1 and 2 over and under the diagonal of 0 enable us to control the parity of 0 blocks. Define F as the set of elementary patterns of size 4 that are not in $\overline{F}$. Then if we denote $\mathbf{T} = \mathbf{T}_F$:

$$\phi_{SA}(\Delta, \mathbf{T}) = \{x \in \{0,1,2\}^{\mathbb{Z}} \ : \ \text{blocks of consecutive 0 are of even length }\}$$

which is not a subshift of finite type as explained in **??**.

B.3. $\mathcal{C}l_{SA}(\mathcal{SFT}) \neq \mathcal{S}ofic$

We construct a subshift of finite type $\mathbf{T}$ such that the projection $\phi_{SA}(\Delta, \mathbf{T})$ on the straight line $y = x$ is not sofic. It is well known that in dimension 1, sofic subshift are exatcly subshift whose language -see definition in **??**- is a regular language [LM95]. The language $\{a^n b^n : n \in \mathbb{N}\}$ is non regular and so we construct $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^2}$ a subshift of finite type and $\pi : \mathcal{A}^{\mathbb{Z}^2} \to \{0, a, b\}^{\mathbb{Z}^2}$ a morphism such that the only allowed patterns in $\mathbf{T}' = \pi(\mathbf{T})$ containing a or b are those of the form $2n \times 2n$:

$$\begin{array}{|cccccc|}\hline &&&&& b\\ &&&& \iddots & \\ &&& b && \\ && a &&& \\ & \iddots &&&& \\ a &&&&& \\ \hline\end{array}$$

The principle is to construct patterns of even size and to localize the center of these patterns to distinguish the a^n part from the b^n part.

Denote $\mathcal{A} = \{*, a, b, 0, 1, 2, 3, 4\}$. The symbols $*, 1, 2, 3$ and 4 help to draw the two diagonals of the square and to distinguish in which quadrant we are. The following set of patterns only allows the construction of even size squares of the form :

$$\begin{array}{|cccccc|}\hline * & 1 & \dots & \dots & 1 & b\\ 4 & \ddots & 1 & 1 & \iddots & 2\\ \vdots & 4 & * & b & 2 & \vdots\\ \vdots & 4 & a & * & 2 & \vdots\\ 4 & \iddots & 3 & 3 & \ddots & 2\\ a & 3 & \dots & \dots & 3 & *\\ \hline\end{array} \quad (*)$$

Let's detail the set of allowed patterns :

$$\text{Squares center : } \begin{array}{|cccc|}\hline * & 1 & 1 & b\\ 4 & * & b & 2\\ 4 & a & * & 2\\ a & 3 & 3 & *\\ \hline\end{array}$$

$$\text{Squares diagonals : } \begin{array}{|ccc|}\hline * & 1 & 1\\ 4 & * & 1\\ 4 & 4 & *\\ \hline\end{array} \quad \begin{array}{|ccc|}\hline 1 & 1 & b\\ 1 & b & 2\\ b & 2 & 2\\ \hline\end{array} \quad \begin{array}{|ccc|}\hline * & 2 & 2\\ 3 & * & 2\\ 3 & 3 & *\\ \hline\end{array} \quad \begin{array}{|ccc|}\hline 4 & 4 & a\\ 4 & a & 3\\ a & 3 & 3\\ \hline\end{array}$$

Squares sides : $\begin{array}{|ccc|}\hline 0 & 0 & 0 \\ 0 & * & 1 \\ 0 & 4 & * \\ \hline\end{array}$ $\begin{array}{|ccc|}\hline 0 & 0 & 0 \\ 1 & 1 & 1 \\ * & 1 & 1 \\ \hline\end{array}$ $\begin{array}{|ccc|}\hline 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ \hline\end{array}$ $\begin{array}{|ccc|}\hline 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & b \\ \hline\end{array}$ $\begin{array}{|ccc|}\hline 0 & 0 & 0 \\ 1 & b & 0 \\ b & 2 & 0 \\ \hline\end{array}$

... and so on for the three other sides.

Squares filling : $\begin{array}{|cc|}\hline 1 & 1 \\ 1 & 1 \\ \hline\end{array}$ $\begin{array}{|cc|}\hline 2 & 2 \\ 2 & 2 \\ \hline\end{array}$ $\begin{array}{|cc|}\hline 3 & 3 \\ 3 & 3 \\ \hline\end{array}$ $\begin{array}{|cc|}\hline 4 & 4 \\ 4 & 4 \\ \hline\end{array}$

Outside the squares : $\boxed{0}$

The only configurations one can construct with these allowed patterns are configurations of $\mathcal{A}^{\mathbb{Z}^2}$ with 0 everywhere except in some places where there are arbitrarily large blocks of the form $(*)$, and the configuration made of the infinite pattern $(*)$. We denote by $\mathbf{T}$ this subshift of finite type.

Let π denote the letter to letter morphism defined by $\pi(x) = 0$ for $x \in \{*, 1, 2, 3, 4\}$ and $\pi(a) = a$, $\pi(b) = b$. Suppose that $\phi_{SA}(\Delta, \mathbf{T})$ is sofic. Since $\mathcal{C}l_F(\mathcal{S}ofic) = \mathcal{S}ofic$ then $\pi(\phi_{SA}(\Delta, \mathbf{T}))$ would also be sofic, which is absurd since:

$$\pi(\phi_{SA}(\Delta, \mathbf{T})) = \{x \in \{0, a, b\}^{\mathbb{Z}} \ : \ \text{all blocks containing only a and b are of the form } a^n b^n\}.$$

So this construction proves that $\mathcal{C}l_{SA}(\mathcal{SFT}) \neq \mathcal{S}ofic$.

Appendix C. Proofs

C.1. Proof of Proposition 1.4

• If $x \in \mathbf{T}$, since $\mathcal{L}(\mathbf{T})$ is the set of all elementary patterns that appears in $\mathbf{T}$, all the elementary patterns of x are in $\mathcal{L}(\mathbf{T})$, that is to say $x \in \mathbf{T}_{\mathcal{L}(\mathbf{T})^c}$. So $\mathbf{T} \subseteq \mathbf{T}_{\mathcal{L}(\mathbf{T})^c}$.

• Reciprocally, let $x \in \mathbf{T}_{\mathcal{L}(\mathbf{T})^c}$. Then every elementary pattern that appears in x is in $\mathcal{L}(\mathbf{T})$. In particular for all $n \in \mathbb{N}$, $x_{\mathbb{S}_n} \in \mathcal{L}(\mathbf{T})$ hence there exists $x_n \in \mathbf{T}$ such that $(x_n)|_{\mathbb{S}_n} = x_{\mathbb{S}_n}$. We have $x_n \to x$, and since $\mathbf{T}$ is a closed set, $x \in \mathbf{T}$.

C.2. Proof of Theorem 2.2

Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ be a recursively enumerable subshift given by P a recursively enumerable set of forbidden patterns. Let $\mathbb{G} < \mathbb{Z}^d$ be a sub-group generated by the vectors $u_1, \dots, u_{d'} \in \mathbb{Z}^d$. The principle of the demonstration is the following : from the set P we construct another recursively enumerable set P' such that $\phi_{SA}(\mathbb{G}, \mathbf{T}) = \mathbf{T}_{P'}$. P' will be the set of all forbidden patterns, that is to say the complementary of the language of $\phi_{SA}(\mathbb{G}, \mathbf{T})$.

We denote by $\mathbf{T}'$ the subshift $\phi_{SA}(\mathbb{G}, \mathbf{T})$, so that $\mathbf{T}' \subseteq \mathcal{A}^{\mathbb{Z}^{d'}}$. We construct a set P' with the following properties : P' is recursively enumerable and $\mathbf{T}' = \mathbf{T}_{P'}$. If p' is a pattern in $\mathcal{A}^{\$_n^{d'}}$, it can be seen as a pattern p in $\mathcal{A}^{\mathbb{U}}$ where $\mathbb{U} = \cup_{i=(i_1,\dots,i_{d'})\in[-n;n]^{d'}} (i_1 u_1 + \dots + i_{d'} u_{d'}) \subseteq \mathbb{Z}^d$ and we denote by $\Phi^{-1}(p')$ the set of elementary patterns in $\mathcal{A}^{\mathbb{Z}^d}$ that contain this pattern p. Let P' be the following set :

$$P' = \bigcup_{n\in\mathbb{N}} \{p' \in \mathcal{A}^{\$_n^{d'}} \; : \; \forall p \in \Phi^{-1}(p'), \exists \tilde{p} \in P \text{ such that } \tilde{p} \text{ appears in } p\}.$$

Lemma C.1. *The set P' is recursively enumerable.*

Proof. Since P is recursively enumerable, there exists a Turing machine $\mathcal{M}$ that accepts every pattern in P and loops on other entries.

Let $\mathcal{M}_{aux}$ be an auxiliary machine with the following behaviour on an entry p : $\mathcal{M}_{aux}$ enumerates all the sub-patterns of p. They are in finite number, we call them $p_1, \dots, p_n$. Then we simulate $\mathcal{M}$ on every p_i, successively one step for each pattern. As soon as $\mathcal{M}$ stops on a p_i, $\mathcal{M}_{aux}$ stops. So this machine loops on p if all its sub-patterns are not in P and stop if there exists a sub-pattern in P.

We construct a Turing machine $\mathcal{M}'$ such that on an entry pattern p', $\mathcal{M}'$ loops if $p' \notin P'$ and stops if $p' \in P'$. Let $p' \in \mathcal{A}^{\mathbb{Z}^{d'}}$ be a pattern. On the entry p' the machine $\mathcal{M}'$ works that way : $\mathcal{M}'$ enumerates all the elementary supports containing the one of p'. We denote this enumeration by $(supp_i)_{i\in\mathbb{N}}$. $\mathcal{M}'$ calculates one step successively on each support $supp_i$:

- Let C_i be the finite set $\mathcal{A}^{supp_i} \cap \Phi^{-1}(p')$. We denote it by $C_i = \{p_1^{(i)}, \dots, p_{k_i}^{(i)}\}$.
- On each $p_k^{(i)}$ successively, $\mathcal{M}'$ simulates $\mathcal{M}_{aux}$.

$\mathcal{M}'$ stops on the entry p' if and only if there exists a support $supp_i$ containing $supp(p')$ such that for all pattern $p \in \Phi^{-1}(p')$ with support $supp_i$, p contains a forbidden pattern in P. This is exactly the definition of P', so P' is recursively enumerable. ∎

Lemma C.2. $\phi_{SA}(\mathbf{T}, \mathbb{G}) = \mathbf{T}_{P'}$.

Proof.

- $\phi_{SA}(\mathbb{G}, \mathbf{T}) \subseteq \mathbf{T}_{P'}$
  Let $y \in \phi_{SA}(\mathbb{G}, \mathbf{T})$. Then there exists $x \in \mathbf{T}$ such that for all $i = (i_1, \dots, i_{d'}) \in \mathbb{Z}^{d'}$, $y_i = x_{i_1 u_1 + \dots + i_{d'} u'_d}$. Let p' be a pattern of y. If p' were in P' then every pattern in C_i would contain a forbidden pattern for $\mathbf{T}$, so in particular x would contain a forbidden pattern for $\mathbf{T}$, that is to say $x \notin \mathbf{T}$ which is absurd. Finally y contains no pattern in P', hence $y \in \mathbf{T}_{P'}$.
- $\mathbf{T}_{P'} \subseteq \phi_{SA}(\mathbb{G}, \mathbf{T})$
  Let $y \in \mathbf{T}_{F'}$. We have $y = (y_i)_{i\in\mathbb{Z}^{d'}}$ with for all $i \in \mathbb{Z}^{d'}$ and $n \in \mathbb{N}$, $y_{i+[-n;n]^{d'}} \notin P'$. For all $n \in \mathbb{N}$ we have $y_{[-n;n]^{d'}} \notin P'$. That means that there exists a pattern $p_n \in \Phi_k^{-1}(y_{[-n;n]^{d'}})$ which contains no forbidden pattern for $\mathbf{T}$. p_n is a finite pattern, we complete it to get a $x_n \in (\mathcal{A} \cup \{\sharp\})^{\mathbb{Z}^d}$, where $\sharp$ is a new symbol not in $\mathcal{A}$. Thus $x_n \in \mathbf{T}_\sharp$ where $\mathbf{T}_\sharp \subseteq (\mathcal{A} \cup \{\sharp\})^{\mathbb{Z}^d}$ is the subshift defined with the same set of forbidden patterns as $\mathbf{T}$. The sequence $(x_n)_{n\in\mathbb{N}}$ is in the compact space $(\mathcal{A} \cup \{\sharp\})^{\mathbb{Z}^d}$ so that we can extract from $(x_n)_{n\in\mathbb{N}}$ a sequence $(x_{\phi(n)})_{n\in\mathbb{N}}$ that converges to $x \in (\mathcal{A} \cup \{\sharp\})^{\mathbb{Z}^d}$. As the x_n are constructed, we also have that $x \in \mathcal{A}^{\mathbb{Z}^d}$ and since $\mathbf{T}_\sharp$ is closed, $x \in \mathbf{T}_\sharp$ hence $x \in \mathbf{T}_\sharp \cap \mathcal{A}^{\mathbb{Z}^d} = \mathbf{T}$. Finally $y_i = x_{i_1 u_1 + \dots + i_{d'} u'_d}$ for all $i = (i_1, \dots, i_{d'}) \in \mathbb{Z}^{d'}$. This proves that $\mathbf{T}_{F'} \subseteq \phi_{SA}(\mathbb{G}, \mathbf{T})$.□

■

The recursively enumerable set P' is such that $\mathbf{T}_{P'} = \phi_{SA}(\mathbb{G}, \mathbf{T})$, so recursively enumerable subshifts are stable under subaction.

C.3. Proof of proposition 3.4

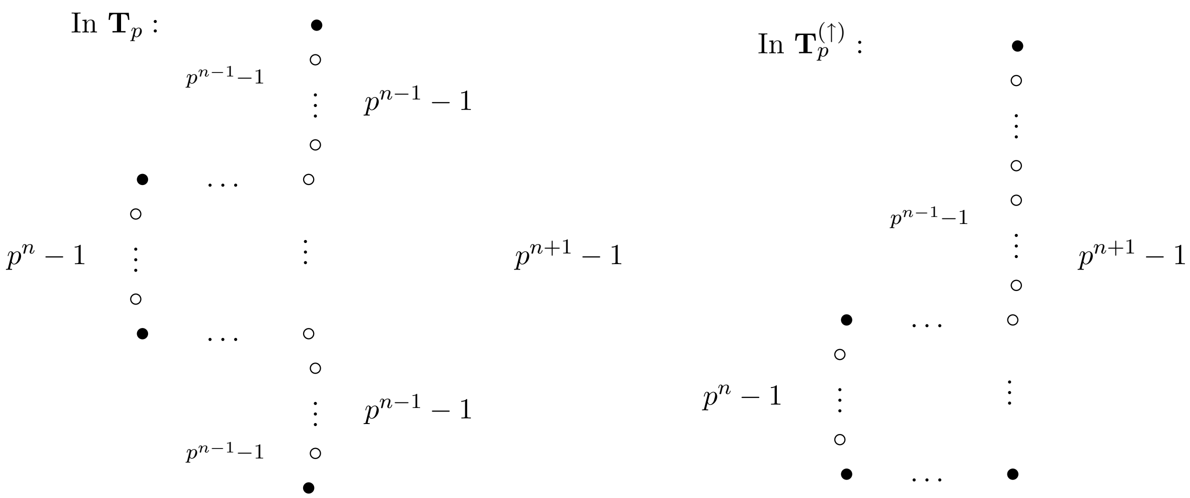
Lemma C.3. *Let p be a prime number and $x \in \mathbf{S}_p$, we denote $E(x) = \{u \in \mathbb{Z}^2 / x_u = \bullet\}$. Then up to a translation :*

$$E(x) = \bigcup_{n=1}^{\infty} \bigcup_{k=1}^{p-1} (p^n\mathbb{Z} + kp^{n-1}) \times (p^n\mathbb{Z} + kp^{n-1})$$

Proof. One can show by induction that this result is true on every elementary support. To have details of this proof see [Hoc07]. ■

A consequence of the structure of $E(x)$ is the following : for every $n \in \mathbb{N}$ there always exists a column and an row in $\mathbf{S}_p$ such that symbols $\bullet$ appear on a p^n-periodic way.

Another consequence : if on a same row -or column- there are two symbols $\bullet$ from a distance p^n, then there exists another row -or column- with two symbols $\bullet$ from a distance p^{n+1}. In $\mathbf{T}_p^{(\uparrow)}$, the symbols $\bullet$ from these two rows -or columns- line up as on the drawing:



Consider a plane $\{i\} \times \mathbb{Z}^2$. Then there exists a colum in $\mathbf{S}_3^{(\uparrow)}$ and a column in $\mathbf{S}_5^{(\uparrow)}$ that define it. On these two columns, symbols $\bullet$ are placed 3^n-periodically and 5^m-periodically respectively. The letters $(\bullet, \bullet)$ of the product alphabet form rectangles of size $3^n \times 5^m$ thanks to rules $(**)$ and $(***)$.

We now focus on such a rectangle of size $3^n \times 5^m$ placed in (i, j, k). Then for all $i_1 = i + 3^{n-1} + \lambda 3^{n+1}$ we have rectangle of width 3^{n+1}. On the same way for all $i_2 = i + \mu 5^m$ we have rectangles of hight 5^m. We look for integers λ and μ such that

$$i + \mu 5^m = i + 3^{n-1} + \lambda 3^{n+1} \Leftrightarrow \mu 5^m = 3^{n-1}(1 + 9\lambda)$$

this is possible since 5^m and 3^n are relatively primes. So there exists i' such that we have a rectangle of size $3^{n+1} \times 5^m$ en (i', j, k). The same kind of reasoning leads to the result of proposition 3.4, since 2,3 and 5 are relatively primes.

C.4. Proof of proposition 4.1

We prove that the relation $\preceq$ is reflexive and transitive.

- $\preceq$ is reflexive : for every language $\mathcal{L}$ we have $\mathcal{L} \preceq \mathcal{L}$. It is sufficient to consider the machine with oracle $\mathcal{L}$ that directly calls for the oracle on the entry word.
- $\preceq$ is transitive. Suppose $\mathcal{L}_1 \preceq \mathcal{L}_2$ and $\mathcal{L}_2 \preceq \mathcal{L}_3$, we prove that $\mathcal{L}_1 \preceq \mathcal{L}_3$. We have at our disposal a machine $\mathcal{M}_2$ with oracle $\mathcal{L}_2$ such that $dom(\mathcal{M}_2) = \mathcal{L}_1$ and a machine $\mathcal{M}_3$ with oracle $\mathcal{L}_3$ such that $dom(\mathcal{M}_3) = \mathcal{L}_2$. We construct a machine $\mathcal{M}$ such that : on an entry p we simulate $\mathcal{M}_2$, and as soon as $\mathcal{M}_2$ calls for its oracle $\mathcal{L}_2$ we simulate $\mathcal{M}_3$. Thus $\mathcal{M}$ stops on p if and only if $p \in \mathcal{L}_1$, and has $\mathcal{L}_3$ for oracle. So $\mathcal{L}_1 \preceq \mathcal{L}_3$.