Simulation of effective subshifts by two-dimensional SFT

N. Aubrun and M. Sablik

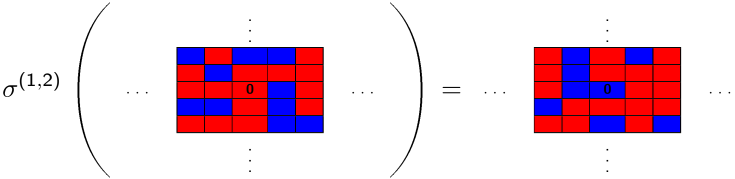
LIGM, Université Paris-Est et LATP, Université de Provence

6 jully 2010

Tilings as dynamical systems

Let $\mathcal{A}$ be a finite alphabet and consider the grid $\mathbb{Z}^d$.
The set $\mathcal{A}^{Z^d}$ endowed by the product toplogy is compact and $\mathbb{Z}^d$ acts by shift on $\mathcal{A}^{\mathbb{Z}^d}$:

$$\forall x \in \mathcal{A}^{\mathbb{Z}^d} \text{ and } \forall n, i \in \mathbb{Z}^d, \sigma^n(x)_i = x_{i+n}$$



The dynamical system $(\mathcal{A}^{Z^d}, \sigma)$ is called the *full shift*.
Let $\mathbf{T} \subset \mathcal{A}^{\mathbb{Z}^d}$ be a closed σ-invariant subset. The dynamical system $(\mathbf{T}, \sigma)$ is called a *subshift*.

Subshifts defined by forbiden patterns

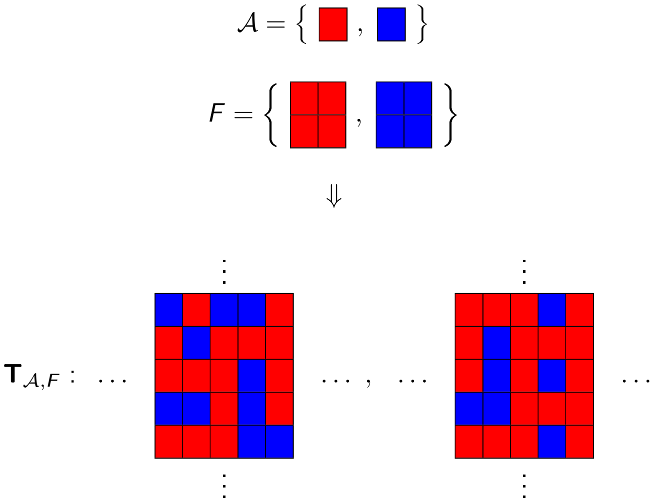
For a finite subset $\mathbb{U} \subset \mathbb{Z}^{\boldsymbol{d}}$ we can consider finite pattern $u \in \mathcal{A}^{\mathbb{U}}$.
Let F be a set of patterns, we define the subshift where the forbidden patterns are F:

$$\mathbf{T}_{\mathcal{A},F} = \{x \in \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}}} / \forall p \in F, p \text{ does not appear in } x\} \subseteq \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}}}$$

Some class of subshifts:

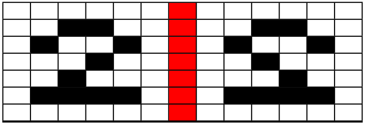
- $\mathbf{T}$ *fullshift* $(\mathbf{T} \in \mathcal{FS}) \Leftrightarrow \Sigma = \mathbf{T}_F = \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}}}$, $F = \emptyset$
- $\mathbf{T}$ *subshift of finite type* $(\mathbf{T} \in \mathcal{SFT}) \Leftrightarrow \exists F$ finite set of patterns such that, $\mathbf{T} = \mathbf{T}_F$
- $\mathbf{T}$ *effective subshift* $(\mathbf{T} \in \mathcal{RE}) \Leftrightarrow \exists F$ recursive enumerable set of patterns such that $\Sigma = \mathbf{T}_F$.

Example of SFT of dimension 2

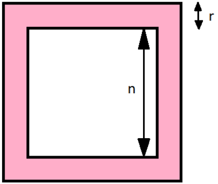


Example of effective subshifts which are not sofic

Let $\mathcal{A} = \{\square, \blacksquare, \blacksquare\}$. Define:

$$\mathbf{T} = \{\square, \blacksquare\}^{\mathbb{Z}^2} \bigcup \left\{ \right.$$  $$\left. \right\}$$

Assume there exists $\Sigma \subset \Lambda^{\mathbb{Z}^2}$ such that $\pi : \Sigma \to \mathbf{T}$ is a factor and $\Sigma \in \mathcal{SFT}$ of order r.
Let n such that $|\Lambda|^{4n \times r + r^2} < 2^{n^2}$.
We can find two different square of $\{\square, \blacksquare\}$ which have the same border.
Thus $\mathbf{T}$ is effective but not sofic.



An effective subshift is not sofic:

- if it is minimal with positive entropy
- if for all configurations, the Kolmogorov complexity of every square verify $\mathcal{K}(\text{square } n \times n) \geq \mathcal{O}(n)$.
- ... other conditions?

Operations on subshifts, simulation

$\mathcal{S}$: set of all subshifts (all dimensions, all finite alphabets)

- *Operations* on subshifts:
  - $op : \mathcal{S} \to \mathcal{S}$
  - or $op : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$
- *Closure* of a class of subshifts $\mathcal{U} \subset \mathcal{S}$ by a set of opérations Op:

  $\mathcal{C}l_{Op}(\mathcal{U})$: smallest subset of $\mathcal{S}$ stable by Op which contains $\mathcal{U}$.

- $\mathbf{T}$ *simule* $\mathbf{T}'$ by Op if $\mathbf{T}' \in \mathcal{C}l_{Op}(\mathbf{T})$ (notation : $\mathbf{T}' \leq_{Op} \mathbf{T}$).

$$\mathcal{C}l_{Op}(\mathbf{T}) = \{\mathbf{T}' \ : \ \mathbf{T}' \leq_{Op} \mathbf{T}\}.$$

Product operation: **Prod**

Definition

Let $\mathbf{T}_i \subseteq \mathcal{A}_i^{\mathbb{Z}^d}$ for any $i \in [1, n]$ be n subshifts of the same dimension, define:

$$\mathbf{Prod}\left(\mathbf{T}_1, \ldots, \mathbf{T}_n\right) = \mathbf{T}_1 \times \cdots \times \mathbf{T}_n \subseteq (\mathcal{A}_1 \times \cdots \times \mathcal{A}_n)^{\mathbb{Z}^d}.$$

$$\left.\begin{array}{c}
\Sigma = \mathbf{T}_F \subseteq \{a, b, c\}^{\mathbb{Z}^2} \\[2ex]
F = \left\{ \begin{array}{|c|c|}\hline * & a \\ \hline a & * \\ \hline \end{array}, \begin{array}{|c|c|}\hline * & b \\ \hline b & * \\ \hline \end{array}, \begin{array}{|c|c|}\hline * & c \\ \hline c & * \\ \hline \end{array} \right\} \\[2ex]
\Sigma' = \mathbf{T}_{F'} \subseteq \{{\color{red}\blacksquare}, {\color{blue}\blacksquare}\}^{\mathbb{Z}^2} \\[2ex]
F' = \left\{ \begin{array}{c} {\color{red}\blacksquare} \\ {\color{blue}\blacksquare} \end{array}, \begin{array}{c} {\color{blue}\blacksquare} \\ {\color{red}\blacksquare} \end{array} \right\}
\end{array}\right\} \Rightarrow
\begin{array}{c}
\Sigma \times \Sigma' \subseteq \{{\color{red}a}, {\color{red}b}, {\color{red}c}, {\color{blue}a}, {\color{blue}b}, {\color{blue}c}\}^{\mathbb{Z}^2} \\[2ex]
F \times \{{\color{red}\blacksquare}, {\color{blue}\blacksquare}\}^{\mathbb{U}} \cup \{a, b, c\}^{\mathbb{U}} \times F' \\[2ex]
\left\{ \begin{array}{|c|c|}\hline {\color{blue}*} & {\color{blue}a} \\ \hline {\color{red}a} & {\color{red}*} \\ \hline \end{array}, \begin{array}{|c|c|}\hline {\color{red}*} & {\color{blue}a} \\ \hline {\color{blue}a} & {\color{red}*} \\ \hline \end{array}, \begin{array}{|c|c|}\hline {\color{red}*} & {\color{red}a} \\ \hline {\color{blue}a} & {\color{blue}*} \\ \hline \end{array}, \ldots \right\}
\end{array}$$

Remark : One has $\mathcal{C}l_{\mathbf{Prod}}(\mathcal{FS}) = \mathcal{FS}$ and $\mathcal{C}l_{\mathbf{Prod}}(\mathcal{SFT}) = \mathcal{SFT}$.

Finite Type operation: **FT**

Définition

Let $\mathbf{T} \in \mathcal{S}$, there exists F' a family of pattern such that $\mathbf{T} = \mathbf{T}_{F'}$.
For a finite family F of pattern, define

$$\mathbf{FT}_F\left(\mathbf{T}\right) = \mathbf{T}_{F \cup F'}.$$

Finite Type operation: **FT**

Définition

Let $\mathbf{T} \in \mathcal{S}$, there exists F' a family of pattern such that $\mathbf{T} = \mathbf{T}_{F'}$.
For a finite family F of pattern, define

$$\mathbf{FT}_F(\mathbf{T}) = \mathbf{T}_{F \cup F'}.$$

Remark : By definition, one has:

$$\mathcal{C}l_{\mathbf{FT}}(\mathcal{FS}) = \mathcal{SFT}$$

Factor operation: **Fact**

Definition

Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}}}$ be a subshift and let $\pi : \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}}} \to \mathcal{B}^{\mathbb{Z}^{\boldsymbol{d}}}$ be a morphism (continous and $\sigma \circ \pi = \pi \circ \sigma$), $\mathbf{Fact}_{\pi}(\mathbf{T}) = \pi(\mathbf{T}) \subseteq \mathcal{B}^{\mathbb{Z}^{\boldsymbol{d}}}$ is a factor of $\mathbf{T}$.

Factor operation: **Fact**

Definition

Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^d}$ be a subshift and let $\pi : \mathcal{A}^{\mathbb{Z}^d} \to \mathcal{B}^{\mathbb{Z}^d}$ be a morphism (continous and $\sigma \circ \pi = \pi \circ \sigma$), $\mathbf{Fact}_\pi(\mathbf{T}) = \pi(\mathbf{T}) \subseteq \mathcal{B}^{\mathbb{Z}^d}$ is a factor of $\mathbf{T}$.

Example :
$\mathcal{A} = \{0, 1, 2\}$ et $\Sigma \subseteq \mathcal{A}^{\mathbb{Z}}$
$\Sigma = \mathbf{T}_{\{00,22,01,12\}}$ et $\pi(0) = \pi(2) = 0, \pi(1) = 1$

$$\Rightarrow \pi(\Sigma) = \{x \in \{0,1\}^{\mathbb{Z}} / \text{ blocks of 0 have even length}\} = \mathbf{T}_{\{0,1\},\{10^{2n+1}1 : n \in \mathbb{N}\}}$$

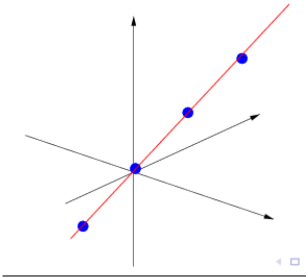
Remark : $SFT \subsetneq \mathcal{C}l_F(SFT)$.
By definition, $\mathcal{C}l_F(SFT)$is the class of *sofic subshifts*. In dimension 1, it is possible to characterize sofic subshift as subshifts with regular forbidden patterns.

Projective Subaction: **SA**

Definition

Let $\mathbb{G}$ be a sub-group of $\mathbb{Z}^d$ finitely generated by $u_1, u_2, \ldots, u_{d'}$ ($d' \leq d$). Let $\mathbf{T} \subseteq \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}}}$ be a subshift :

$$\mathbf{SA}_{\mathbb{G}}(\mathbf{T}) = \left\{ \begin{array}{cc} y \in \mathcal{A}^{\mathbb{Z}^{\boldsymbol{d}'}} & : \exists x \in \mathbf{T} \text{ such that } \forall i_1, \ldots, i_{d'} \in \mathbb{Z}^{d'}, \\ & y_{i_1,\ldots,i_{d'}} = x_{i_1 u_1 + \cdots + i_{d'} u_{d'}} \end{array} \right\}.$$



Example of projective subaction

Let $\mathcal{A}=\{0,1,2,3,4,a,b\}$, it is possible to define a SFT $\mathbf{T}\subset\mathcal{A}^{\mathbb{Z}^2}$ such that 0 is a background and finite non 0composant have the following form:

$$\boxed{\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & * & 1 & \dots & \dots & 1 & b & 0\\
0 & 4 & \ddots & 1 & 1 & \iddots & 2 & 0\\
0 & \vdots & 4 & * & b & 2 & \vdots & 0\\
0 & \vdots & 4 & a & * & 2 & \vdots & 0\\
0 & 4 & \iddots & 3 & 3 & \ddots & 2 & 0\\
0 & a & 3 & \dots & \dots & 3 & * & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}}$$

Let $\mathbb{G}=\{(i,i):i\in\mathbb{Z}\}$, we have:

$$\mathbf{SA}_{\mathbb{G}}(\mathbf{T})=\mathbf{T}_{\{*a^n b^m *:n\neq m\}}\notin\mathcal{S}ofic.$$

Proposition

$\mathcal{C}l_{\mathbf{SA}}(\mathcal{RE})=\mathcal{RE}$

In particular $\mathcal{C}l_{\mathbf{SA}}(\mathcal{S}ofic)=\mathcal{C}l_{\mathbf{Fact},\mathbf{SA}}(\mathcal{SFT})\subset\mathcal{RE}$

Main result

Theorem (Hochman)

Every effective subshifts of dimension d can be obtained using **SA** and **Fact** on a SFT of dimension $d+2$.

$$\mathcal{C}l_{\textbf{Fact},\textbf{SA}}(\mathcal{SFT}\cap\mathcal{S}_{d+2})\cap\mathcal{S}_{\leq d}=\mathcal{RE}\cap\mathcal{S}_{\leq d}$$

Main result

Theorem (Hochman)

Every effective subshifts of dimension d can be obtained using **SA** and **Fact** on a SFT of dimension $d+2$.

$$\mathcal{C}l_{\textbf{Fact},\textbf{SA}}(\mathcal{SFT} \cap \mathcal{S}_{d+2}) \cap \mathcal{S}_{\leq d} = \mathcal{RE} \cap \mathcal{S}_{\leq d}$$

Theorem (Aubrun & S. or Durand & Romashchenko & Shen)

Every effective subshifts of dimension d can be obtained using **SA** and **Fact** on a SFT of dimension $d+1$.

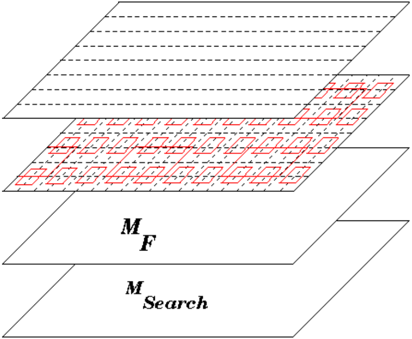
$$\mathcal{C}l_{\textbf{Fact},\textbf{SA}}(\mathcal{SFT} \cap \mathcal{S}_{d+1}) \cap \mathcal{S}_{\leq d} = \mathcal{RE} \cap \mathcal{S}_{\leq d}$$

Four layers

We want to simulate an effective subshift $\Sigma \subseteq {\mathcal{A}_\Sigma}^{\mathbb{Z}}$ with a SFT of dimension 2 (it is easy to generalize to other dimensions).

Different layers:

- **Layer 1**: a superposition of configuration $x \in {\mathcal{A}_\Sigma}^{\mathbb{Z}}$ (a candidate $x \in \Sigma$?),
- **Layer 2**: computation zones used by $\mathcal{M}_{\texttt{Forbid}}$ et $\mathcal{M}_{\texttt{Search}}$ equiped with a clock,
- **Layer 3**: Turing machine $\mathcal{M}_{\texttt{Forbid}}$ enumerates forbidden patterns,
- **Layer 4**: Turing machine $\mathcal{M}_{\texttt{Search}}$ checks the configuration x.



Layer 1: the candidate $x \in \Sigma$?

We align all letters of the first layer to obtain the same configuration horizontally.
Finite condition:

$$\mathbf{Align} = \left\{ \begin{array}{|c|}\hline a \\ \hline b \\ \hline \end{array}, a \neq b \in \mathcal{A}_\Sigma \right\}$$

The first layer is constitued by:

$$\mathbf{T_{Align}} = \mathbf{FT_{Align}}\left(\mathcal{A}_\Sigma^{\mathbb{Z}^2}\right)$$

Aim:

We want to eliminate each x which contains forbidden patterns of Σ.

We want to code a Turing machine with a SFT

Definition: Turing machine

A *Turing machine* is $\mathcal{M} = (Q, \Gamma, \sharp, q_0, \delta, Q_F)$ with:

- Q fite set of states; $q_0 \in Q$ initial state;
- Γ finite alphabet;
- $\sharp \notin \Gamma$ white symbol
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, \cdot, \rightarrow\}$ fonction of transition;
- $F \subset Q_F$ set of final states.

The rule $\delta(q_1, x) = (q_2, y, \leftarrow)$ is coded by:

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

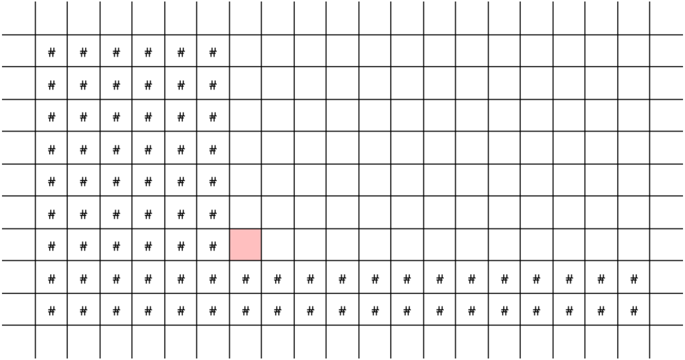
It is possible to consider the SFT $\mathbf{T}_{\mathcal{M}}$ for a Turing machine $\mathcal{M}$

Problem

How initialize the computation?

Initialisation of the computation

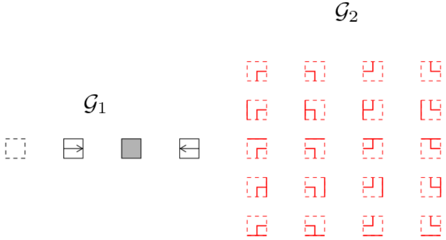
If a particular tile appear just one time, it is easy to initialize the computation by finite type conditions :



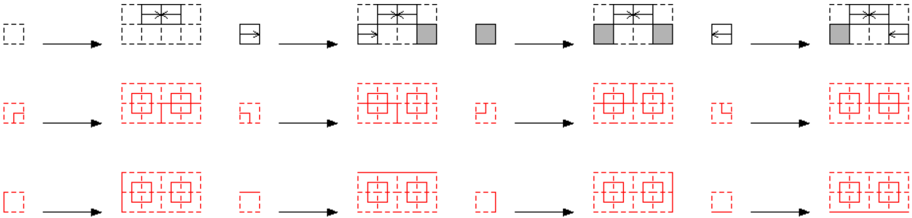
But, by compacity, it is impossible to force the presence of a unique tile.

Layer 2: the grid

The alphabets $\mathcal{G}_1 \times \mathcal{G}_2$ on which the substitution is defined

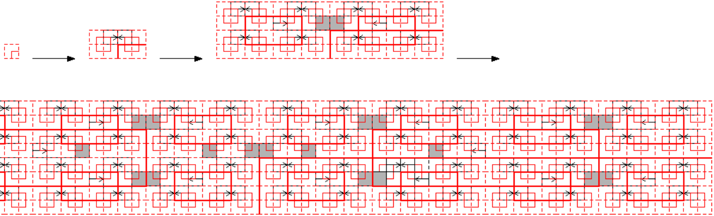


Substitution rules:



Layer 2: the grid

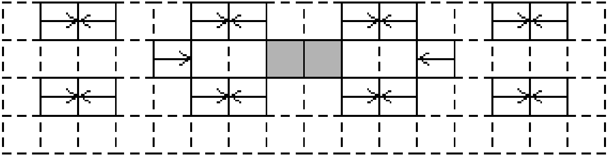
Four iterations of the substitution $s_{\texttt{Grid}}$:



By the result of Mozes, the fixe point obtained generates a sofic denoted $\mathbf{T}_{\texttt{Grid}}$.

Description of computation zones

We just consider the projection on $\mathcal{G}_1$.

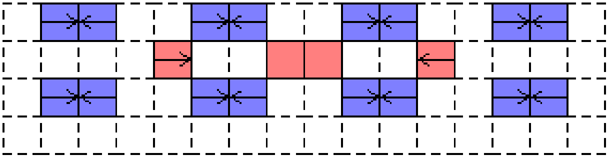


$\square$: tile of communication

$\boxminus$, $\boxminus$, $\blacksquare$: tiles of computation

Description of computation zones

We just consider the projection on $\mathcal{G}_1$.

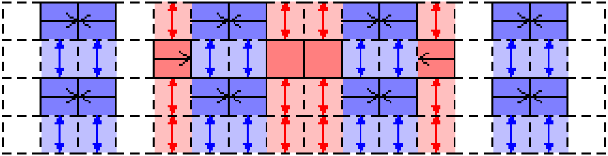


$\square$: tile of communication

$\boxminus$, $\boxminus$, $\blacksquare$: tiles of computation

Description of computation zones

We just consider the projection on $\mathcal{G}_1$.

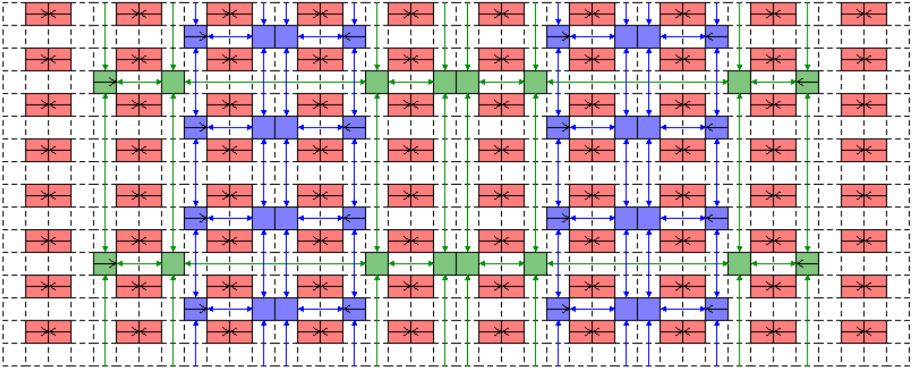


$\square$: tile of communication

$\boxed{\rightarrow}$, $\boxed{\leftarrow}$, $\blacksquare$: tiles of computation

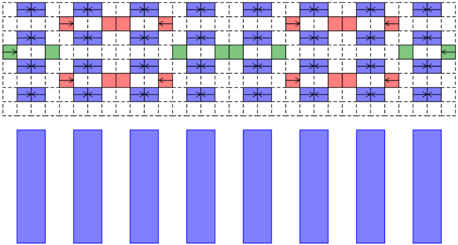
Description of computation zones

And more generally:



Description of computation zones

We have fractionated computation strips of different level (level 1, level 2, level 3):

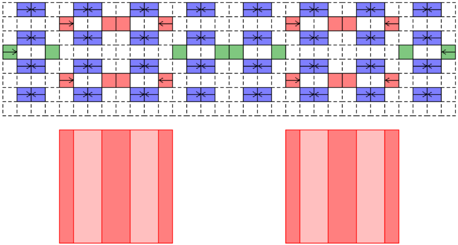


A computation strip of level n has the following properties :

- the width is 2^n,
- each rows have computation boxes of the same level,
- there is a computation line every 2^n lines.

Description of computation zones

We have fractionated computation strips of different level (level 1, level 2, level 3):

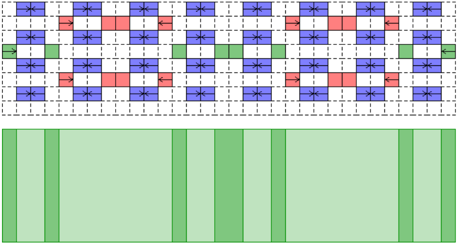


A computation strip of level n has the following properties :

- the width is 2^n,
- each rows have computation boxes of the same level,
- there is a computation line every 2^n lines.

Description of computation zones

We have fractionated computation strips of different level (level 1, level 2, level 3):

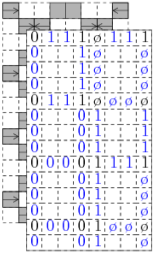


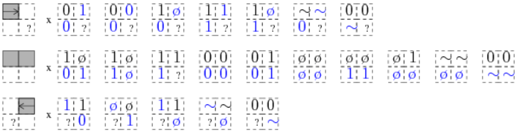
A computation strip of level n has the following properties :

- the width is 2^n,
- each rows have computation boxes of the same level,
- there is a computation line every 2^n lines.

Layer 2: A grid and a clock

To initialize the computation we code a clock by finite rules:





The layer 2 can be defined as:

$$\mathbf{T}_{\texttt{Clock}} = \mathbf{FT}_{\mathbf{Count} \cup \mathbf{Consist} \cup \mathbf{Synchro}} \left(\mathbf{Prod} \left(\mathbf{T}_{\texttt{Grid}}, \mathcal{C}^{\mathbb{Z}^2} \right) \right)$$

For a strip of level n, this allows to initialize computation every 2^{2^n}.

How to simulate a Turing machine with $\mathbf{T}_{\texttt{Clock}}$

We use the operation **FT** on **Prod** $\left(\mathbf{T}_{\texttt{Clock}}, \mathcal{A}^2_{\mathcal{M}}\right)$ to add computation of $\mathcal{M}$ in the strips:

- conditions **Init** : when the clock $= 0$, the empty word is written on the comptation zone and the initial state q_0 appears at the beggining (symbol $\boxed{\rightarrow}$);

How to simulate a Turing machine with $\mathbf{T}_{\texttt{Clock}}$

We use the operation **FT** on **Prod** $\left(\mathbf{T}_{\texttt{Clock}}, \mathcal{A}^2_{\mathcal{M}}\right)$ to add computation of $\mathcal{M}$ in the strips:

- conditions **Init** : when the clock $= 0$, the empty word is written on the comptation zone and the initial state q_0 appears at the beggining (symbol $\boxminus$);
- conditions **Comp** : when the clock $\neq 0$, we use the rules of $\mathbf{T}_{\mathcal{M}}$;

How to simulate a Turing machine with $\mathbf{T}_{\texttt{Clock}}$

We use the operation **FT** on **Prod** $\left(\mathbf{T}_{\texttt{Clock}}, \mathcal{A}^2_{\mathcal{M}}\right)$ to add computation of $\mathcal{M}$ in the strips:

- conditions **Init** : when the clock $= 0$, the empty word is written on the comptation zone and the initial state q_0 appears at the beggining (symbol $\boxminus$);
- conditions **Comp** : when the clock $\neq 0$, we use the rules of $\mathbf{T}_{\mathcal{M}}$;
- conditions **Transfer** : if the computation is in a communication zone, the states of the Turing machine is transfered (horizontally and vertically);

How to simulate a Turing machine with $\mathbf{T}_{\texttt{Clock}}$

We use the operation **FT** on **Prod** $\left(\mathbf{T}_{\texttt{Clock}}, \mathcal{A}^2_{\mathcal{M}}\right)$ to add computation of $\mathcal{M}$ in the strips:

- conditions **Init** : when the clock $= 0$, the empty word is written on the comptation zone and the initial state q_0 appears at the beggining (symbol $\boxminus$);
- conditions **Comp** : when the clock $\neq 0$, we use the rules of $\mathbf{T}_{\mathcal{M}}$;
- conditions **Transfer** : if the computation is in a communication zone, the states of the Turing machine is transfered (horizontally and vertically);
- conditions **Bound** : if the computation want to go out the computation zone, the computation stop.

How to simulate a Turing machine with $\mathbf{T}_{\texttt{Clock}}$

We use the operation **FT** on **Prod** $\left(\mathbf{T}_{\texttt{Clock}}, \mathcal{A}^2_{\mathcal{M}}\right)$ to add computation of $\mathcal{M}$ in the strips:

- conditions **Init** : when the clock $= 0$, the empty word is written on the comptation zone and the initial state q_0 appears at the beggining (symbol $\boxminus$);
- conditions **Comp** : when the clock $\neq 0$, we use the rules of $\mathbf{T}_{\mathcal{M}}$;
- conditions **Transfer** : if the computation is in a communication zone, the states of the Turing machine is transfered (horizontally and vertically);
- conditions **Bound** : if the computation want to go out the computation zone, the computation stop.

For a Turing machine $\mathcal{M}$, one can consider:

$$\mathbf{T}_{\mathcal{M}} = \mathbf{FT}_{\mathbf{Work}_{\mathcal{M}}}\left(\mathbf{Prod}\left(\mathbf{T}_{\texttt{Grid}}, \mathcal{A}^{\mathbb{Z}^2}_{\mathcal{M}}\right)\right)$$

where $\mathbf{Work}_{\mathcal{M}} = \mathbf{Transfer} \cup \mathbf{Init} \cup \mathbf{Comp} \cup \mathbf{Bound}$.

How to simulate a Turing machine with $\mathbf{T}_{\texttt{Clock}}$

$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a
$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	a	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	b	$(q_{\Vert},\Vert)$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\sharp$	$\uparrow$	$\uparrow$
$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a
a	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$(q_{b+},\sharp)$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\sharp$	$\uparrow \leftrightarrow$
$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a
$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	a	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	b	$(q_{\Vert},\sharp)$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\sharp$	$\uparrow$	$\uparrow$
$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a
$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a	$(q_{b+},\sharp)$	$\uparrow$	$\uparrow$	a
$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	a	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$(q_{b+},\sharp)$	$\sharp$	$\uparrow \leftrightarrow$	$\uparrow \leftrightarrow$	$\sharp$	$\uparrow$	$\uparrow$
$\uparrow$	$(q_0,\sharp)$	$\sharp$	$\uparrow$		$(q_0,\sharp)$	$\sharp$			$(q_0,\sharp)$	$\sharp$		$\uparrow$	$(q_0$
$(q_0,\sharp)$	$\leftrightarrow$	$\leftrightarrow$	$\sharp$	$\leftrightarrow$	$\leftrightarrow$	$\leftrightarrow$	$\leftrightarrow$	$\leftrightarrow$	$\leftrightarrow$	$\leftrightarrow$	$\leftrightarrow$	$\sharp$	$\leftrightarrow$

Layer 3: Detection of forbidden patterns by $\mathcal{M}_{\texttt{Forbid}}$

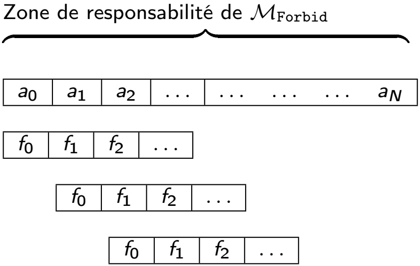
- $\mathcal{M}_{\texttt{Forbid}}$ generates forbidden patterns for Σ

Layer 3: Detection of forbidden patterns by $\mathcal{M}_{\texttt{Forbid}}$

- $\mathcal{M}_{\texttt{Forbid}}$ generates forbidden patterns for Σ
- each strip has a responsibility zone and $\mathcal{M}_{\texttt{Forbid}}$ checks if a forbidden pattern appears in this zone;

Layer 3: Detection of forbidden patterns by $\mathcal{M}_{\texttt{Forbid}}$

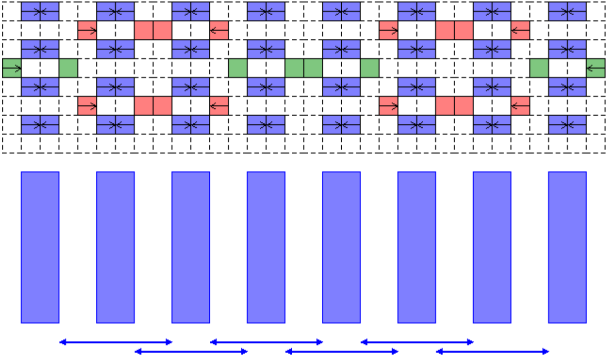
- $\mathcal{M}_{\texttt{Forbid}}$ generates forbidden patterns for Σ
- each strip has a responsibility zone and $\mathcal{M}_{\texttt{Forbid}}$ checks if a forbidden pattern appears in this zone;



- to obtain the value in the layer 1 of a_k, $\mathcal{M}_{\texttt{Forbid}}$ need the help of $\mathcal{M}_{\texttt{Search}}$: $\mathcal{M}_{\texttt{Forbid}}$ gives the address k and receive a_k

Responsability zones of $\mathcal{M}_{\texttt{Forbid}}$

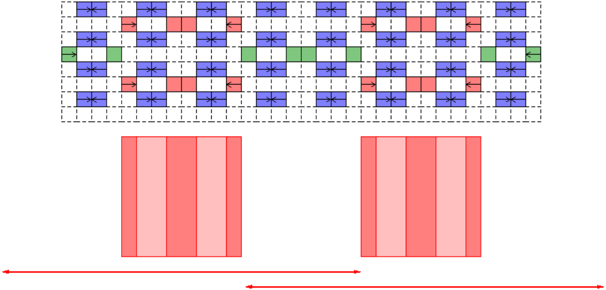
Responsability zones must scan all the configuration considered, thus responsability zones of same level overlap ($\ldots x_{-2}x_{-1}x_0 \in \mathcal{L}(\Sigma)$ and $x_0x_1x_2\cdots \in \mathcal{L}(\Sigma)$ does not imply that $\ldots x_{-2}x_{-1}x_0x_1x_2\cdots \in \mathcal{L}(\Sigma)$).



The size of a responsability zone of level n is $6 \times 4^{n-1}$.

Responsability zones of $\mathcal{M}_{\texttt{Forbid}}$

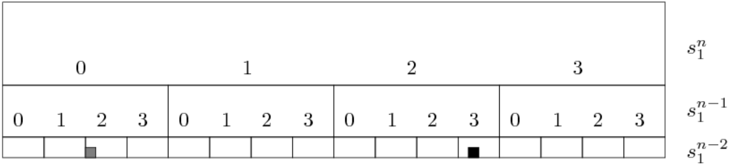
Responsability zones must scan all the configuration considered, thus responsability zones of same level overlap ($\ldots x_{-2}x_{-1}x_0 \in \mathcal{L}(\Sigma)$ and $x_0x_1x_2\cdots \in \mathcal{L}(\Sigma)$ does not imply that $\ldots x_{-2}x_{-1}x_0x_1x_2\cdots \in \mathcal{L}(\Sigma)$).



The size of a responsability zone of level n is $6 \times 4^{n-1}$.
The Turing machine $\mathcal{M}_{\texttt{Forbid}}$ of a level n can ask at $\mathcal{M}_{\texttt{Search}}$ of the same level or neighbor $\mathcal{M}_{\texttt{Search}}$ of the same level.

Layer 4: The Turing machine $\mathcal{M}_{\texttt{Search}}$

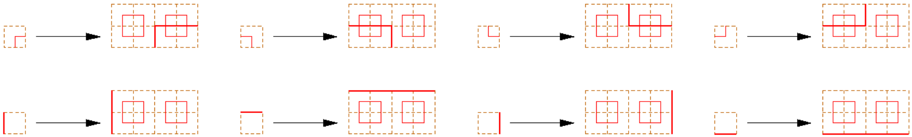
We want to construct a Turing machine $\mathcal{M}_{\texttt{Search}}$ which can comunicate with different levels in view to explore all the configuration cheaked.
When you give a strip it is possible to give an address of each boxes in function of the place in a de-substitute pattern:



The black box address is 231 and the grey box address is 020.

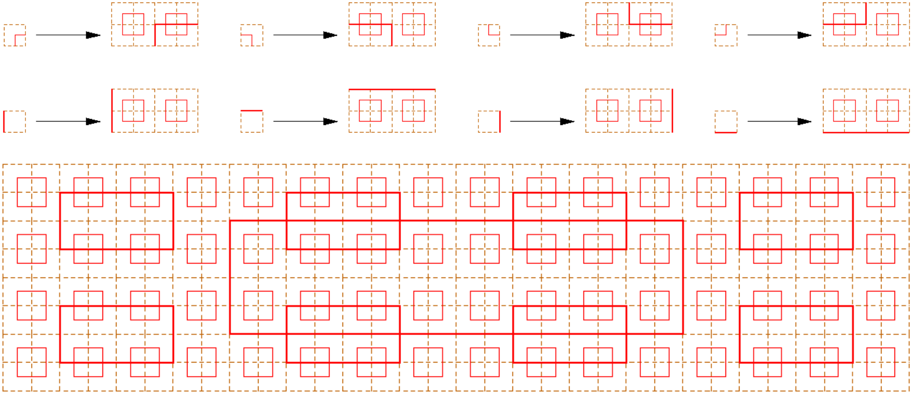
Communication between $\mathcal{M}_{\texttt{Search}}$ of different level

With the alphabet $\mathcal{G}_2$, we construct channels of communication:



Communication between $\mathcal{M}_{\mathtt{Search}}$ of different level

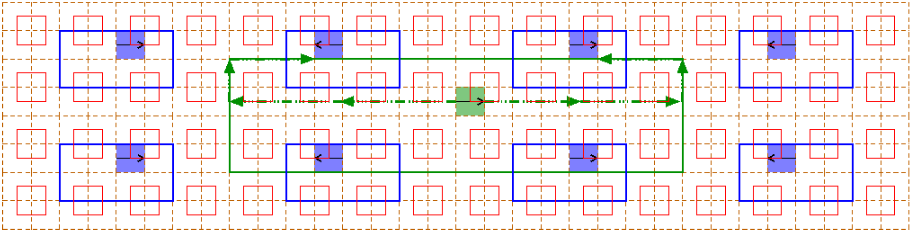
With the alphabet $\mathcal{G}_2$, we construct channels of communication:



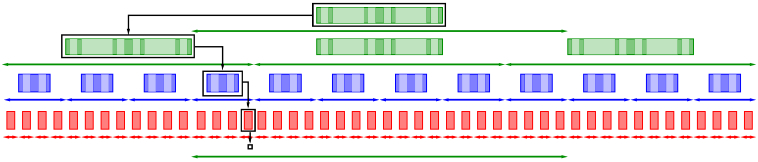
Communication between $\mathcal{M}_{\mathtt{Search}}$ of different level

Principle :

- each border of a computation zone ($\boxminus\!\rightarrow$ or $\leftarrow\!\boxminus$) is in the center of a rectangle of level n;
- for each rectangle of level n, there is in connection only with $\boxminus\!\rightarrow$ and $\leftarrow\!\boxminus$ of two Turing machines of level $n-1$

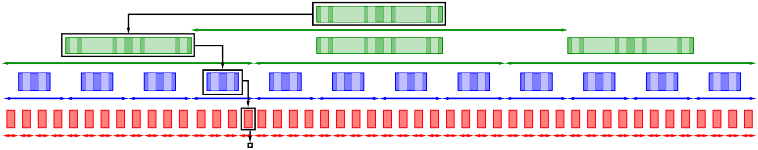


$\mathcal{M}_{\texttt{Search}}$ holds



- Each $\mathcal{M}_{\texttt{Search}}$ has enough space to code address:
  - the size of computation zone of level n is 2^n,
  - the size of a responsability zone of level n is $6 \times 4^{n-1}$,
  - with an alphabet of cardinality 4, we just need n bits.

$\mathcal{M}_{\texttt{Search}}$ holds



- Each $\mathcal{M}_{\texttt{Search}}$ has enough space to code address:
  - the size of computation zone of level n is 2^n,
  - the size of a responsability zone of level n is $6 \times 4^{n-1}$,
  - with an alphabet of cardinality 4, we just need n bits.
- Each $\mathcal{M}_{\texttt{Search}}$ can calculate for a given $\mathcal{M}_{\texttt{Forbid}}$:
  - let $t(n)$ be the time taken by $\mathcal{M}_{\texttt{Search}}$ to give an answer to a Turing machine $\mathcal{M}_{\texttt{Forbid}}$ of order n. One has

    $$t(n) \leq n \times 2^n + 4 \times t(n-1)$$

    So $t(n) \leq 2^n \times \mathcal{O}(n^2 2^n)$.
  - a clock of level n is initialised every 2^{2^n}
  - the time of answer is "absorbed" by the exponential time of the clock.

The final construction

- condition **Request** : $\mathcal{M}_F$ ask $\mathcal{M}_{\mathsf{Search}}$ the value of a box in the respnsability zone and wait the answer
- condition **Forbid** : we exclude the configuration when $\mathcal{M}_{\mathtt{Forbid}}$ encounter a forbidden pattern

$$\mathbf{T}_{\mathtt{Final}} = \mathbf{FT}_{\mathbf{Work}_{\mathcal{M}_{\mathtt{Request}}} \cup \mathbf{Work}_{\mathcal{M}_{\mathtt{Search}}} \cup \mathbf{Com} \cup \mathbf{Forbid}} \left(\mathbf{T}_{\mathtt{Layer}}\right) .$$

where

$$\mathbf{T}_{\mathtt{Layer}} = \mathbf{Prod}\left(\mathbf{FT}_{\mathbf{Align}}\left(\mathcal{A}^{\mathbb{Z}^2}\right), \mathbf{T}_{\mathtt{Clock}}, \mathcal{A}^{\mathbb{Z}^2}_{\mathcal{M}_{\mathtt{Forbid}}}, \mathcal{A}^{\mathbb{Z}^2}_{\mathcal{M}_{\mathtt{Search}}}\right)$$

To obtain Σ :

- operation **Fact** to keep only letters of $\mathcal{A}_\Sigma$
- operation **SA** to keep only an horizontal line

Application 1: An order on subshifts

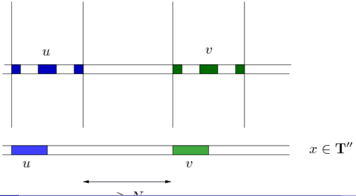
A *Turing machine with semi-oracle $\mathcal{L}$* has the following behaviour: the machine reads an entry pattern p and writes a pattern on the oracle tape, until the state $q_?$ is reached. If the pattern written on the oracle tape is in $\mathcal{L}$ then the machine stops, else it keeps on calculating.
We can define the following semi-order:

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

Theorem

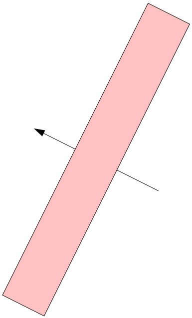
Let $\mathbf{T}$ be a strongly specified subshift, one has:

$$\mathcal{C}l_{\mathbf{Prod},\mathbf{Fact},\mathbf{FT},\mathbf{SA},\mathbf{SE}}(\mathbf{T}) = \{\mathbf{T}_{\mathcal{L}} : \mathcal{L} \preceq \mathcal{L}(\mathbf{T})^c\}.$$



Application 2: Expansive directions

Let $\Sigma \subset \mathcal{A}^{\mathbb{Z}^2}$ be a subshift. Σ is *expansive* according a line Δ if two configurations $x, y \in \Sigma$ are similar around Δ, then $x = y$.



Theorem

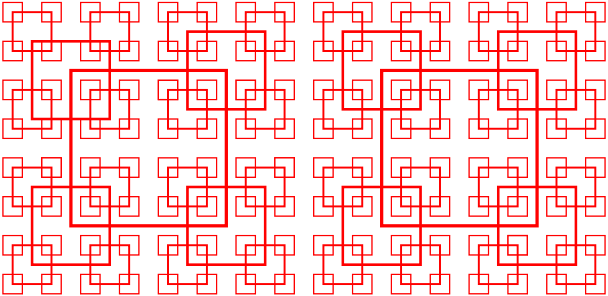
For an effective subshift, the bounds of expansive cone are exactely given by effective number.

What happen for sofic subshift?

Application 3: Effective multidimensional S-adic systems

Let $s_1 : \boxed{*} \to \begin{array}{|c|c|}\hline * & * \\ \hline * & * \\ \hline\end{array}$ and $s_2 : \boxed{*} \to \begin{array}{|c|c|}\hline * & * \\ \hline * & * \\ \hline\end{array}$ be two square substitutions.
Let $(\tau_i)_{\mathbb{N}}$ be an effective sequence of $\{0;1\}$.
Define the effective multidimensionel $\{s_1; s_2\}$-adic systems following τ:

$$\mathbf{T} = \{x \in \mathcal{A}^{\mathbb{Z}^2} : \text{admisible patterns are } s_{\tau_0} \circ s_{\tau_1} \circ \cdots \circ s_{\tau_{k-1}} \circ s_{\tau_k}(\square)\}$$



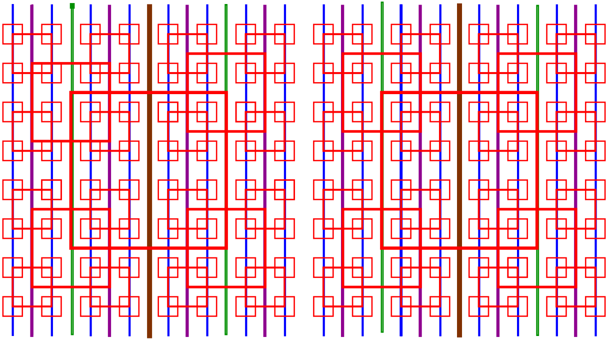
Theorem

Effective multidimensional S-adic systems are sofic.

Application 3: Effective multidimensional S-adic systems

Let $s_1 : \boxed{*} \rightarrow \begin{array}{|c|c|}\hline * & * \\ \hline * & * \\ \hline\end{array}$ and $s_2 : \boxed{*} \rightarrow \begin{array}{|c|c|}\hline * & * \\ \hline * & * \\ \hline\end{array}$ be two square substitutions.
Let $(\tau_i)_{\mathbb{N}}$ be an effective sequence of $\{0;1\}$.
Define the effective multidimensionel $\{s_1;s_2\}$-adic systems following τ:

$$\mathbf{T} = \{x \in \mathcal{A}^{\mathbb{Z}^2} : \text{admisible patterns are } s_{\tau_0} \circ s_{\tau_1} \circ \cdots \circ s_{\tau_{k-1}} \circ s_{\tau_k}(\square)\}$$



Theorem

Effective multidimensional S-adic systems are sofic.

Application 4: Discrete plane