Fonctions de complexité et complexité tout court

Pascal Vanier, joint work with Ronnie Pavlov

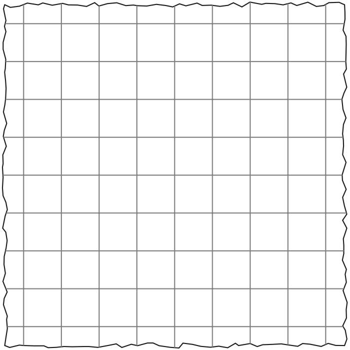
Laboratoire d'Algorithmique Complexité et Logique, UPEC

Codys 2019

Subshifts and subshifts of finite type

A **finite** alphabet:

$$\Sigma = \{\square, \square\}$$

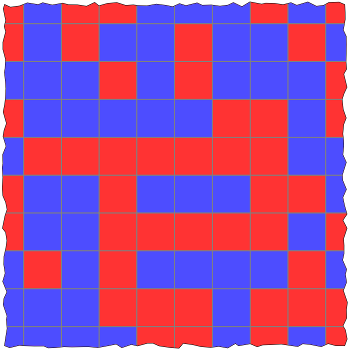


Subshifts and subshifts of finite type

A finite alphabet:

$\Sigma = \{$  ,  $\}$

A tiling or configuration is a coloring of $\mathbb{Z}^d$:



Subshifts and subshifts of finite type

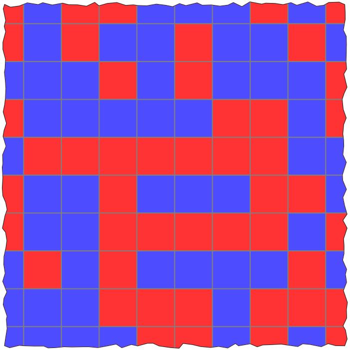
A finite alphabet:

$$\Sigma = \{\ \}$$

A finite number of forbidden patterns:

$$\mathcal{F} = \{\ \}$$

A tiling or configuration is a coloring of $\mathbb{Z}^d$:

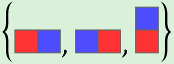


Subshifts and subshifts of finite type

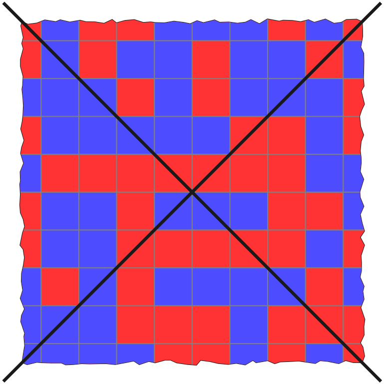
A **finite** alphabet:

$\Sigma =$ 

A **finite** number of forbidden patterns:

$\mathcal{F} =$ 

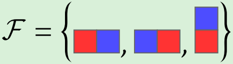
A **tiling** or **configuration** is a coloring of $\mathbb{Z}^d$:



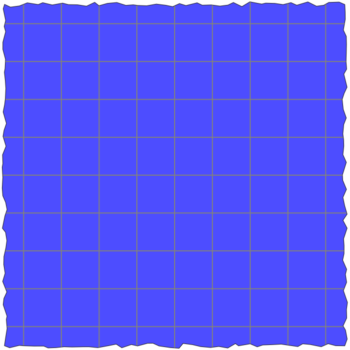
Subshifts and subshifts of finite type

A **finite** alphabet:

A **finite** number of forbidden patterns:



A **tiling** or **configuration** is a coloring of $\mathbb{Z}^d$:



Subshifts and subshifts of finite type

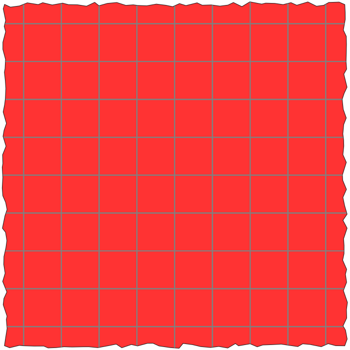
A **finite** alphabet:

$$\Sigma = \{\ \}$$

A **finite** number of forbidden patterns:

$$\mathcal{F} = \{\ \}$$

A **tiling** or **configuration** is a coloring of $\mathbb{Z}^d$:

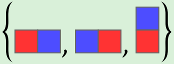


Subshifts and subshifts of finite type

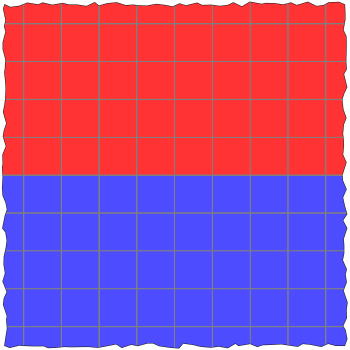
A **finite** alphabet:

$\Sigma =$ 

A **finite** number of forbidden patterns:

$\mathcal{F} =$ 

A **tiling** or **configuration** is a coloring of $\mathbb{Z}^d$:



Subshifts and subshifts of finite type

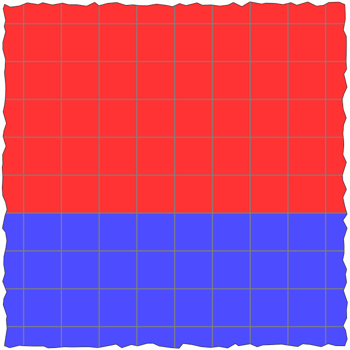
A **finite** alphabet:

$\Sigma = \{$  $\}$

A **finite** number of forbidden patterns:

$\mathcal{F} = \{$  $\}$

A **tiling** or **configuration** is a coloring of $\mathbb{Z}^d$:



Subshifts and subshifts of finite type

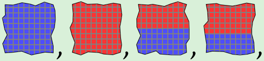
A **finite** alphabet:

$$\Sigma = \{$$  $$\}$$

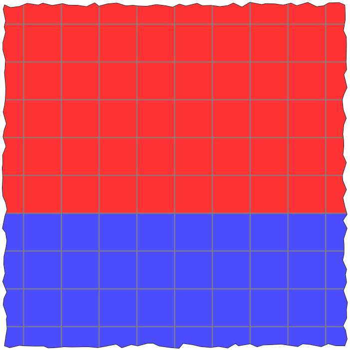
A **finite** number of forbidden patterns:

$$\mathcal{F} = \{$$  $$\}$$

Subshift of finite type (SFT): set of configurations avoiding $\mathcal{F}$. We note $\mathcal{X}_{\mathcal{F}}$:

$$\mathcal{X}_{\mathcal{F}} = \{$$  $$, \ldots \}$$

A **tiling** or **configuration** is a coloring of $\mathbb{Z}^d$:



Subshifts and subshifts of finite type

A finite alphabet:

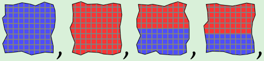
$$\Sigma = \{ \quad , \quad \}$$

A finite number of forbidden patterns:

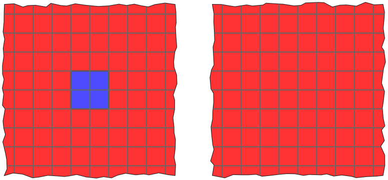
$$\mathcal{F} = \left\{ \quad , \quad , \quad \right\}$$

Subshift of finite type (SFT): set of configurations avoiding $\mathcal{F}$. We note $\mathcal{X}_{\mathcal{F}}$:

$$\mathcal{X}_{\mathcal{F}} = \left\{ \quad , \quad , \quad , \quad , \ldots \right\}$$



The family my also be infinite we then talk about subshifts.



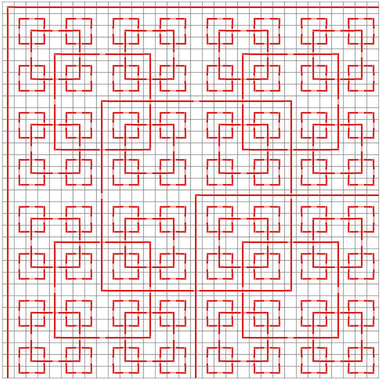
Things get interesting in $d \geq 2$

[Berger 1964] There exists an SFT containing only **non-periodic points**.

[Berger 1964]

Things get interesting in $d \geq 2$

[Berger 1964] There exists an SFT containing only **non-periodic points**.



[Robinson 1971]

Things get interesting in $d \geq 2$

[Berger 1964] There exists an SFT containing only **non-periodic points**.

And numerous others:

[Knuth 1968]

[Anderaa & Lewis 1974]

[Kari 1996]

[Ollinger 2008]

[Durand, Romashchenko & Shen 2008]

[Poupet 2010]

[Jeandel & Rao 2015]

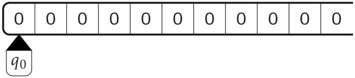
...

Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.

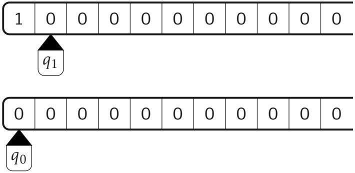
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



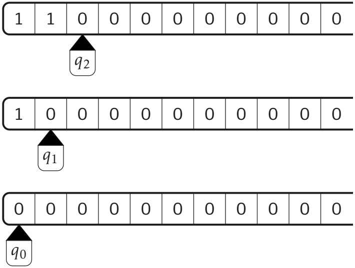
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



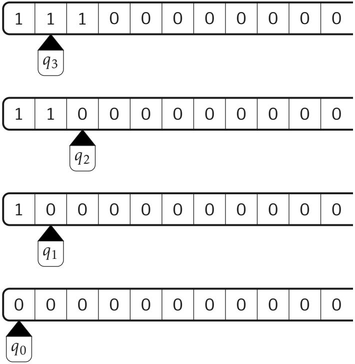
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



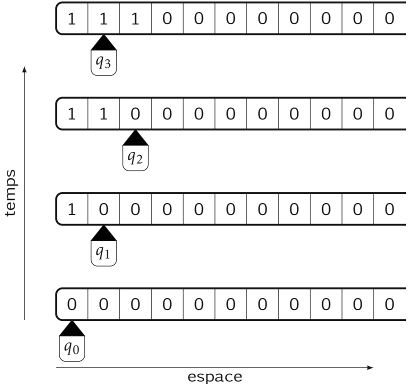
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



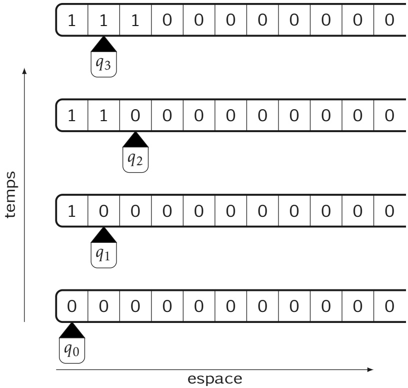
Nothing is easy in $d \geq 2$

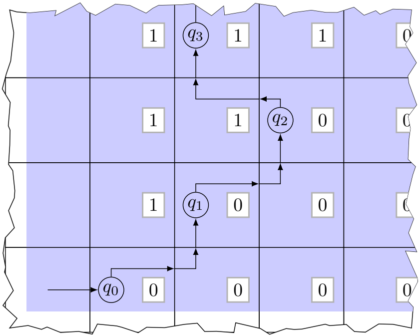
Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



Nothing is easy in $d \geq 2$

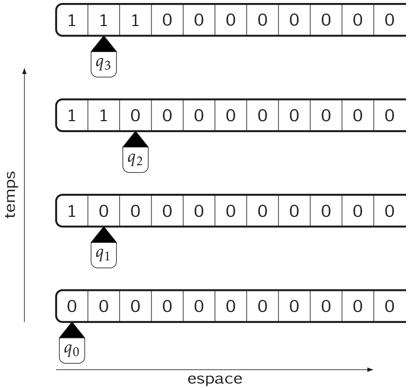
Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



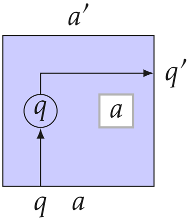


Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_\mathcal{F}$ is empty, given $\mathcal{F}$ as input.

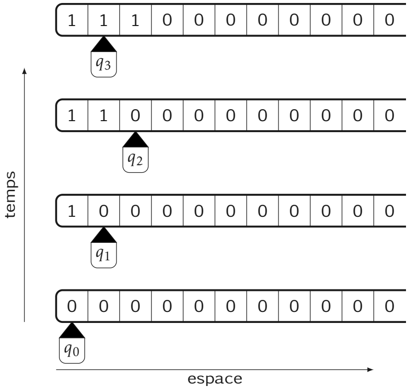


$$(q, a) \longrightarrow (q', a', \rightarrow)$$

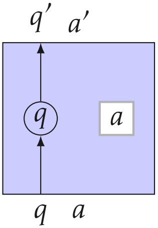


Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_\mathcal{F}$ is empty, given $\mathcal{F}$ as input.

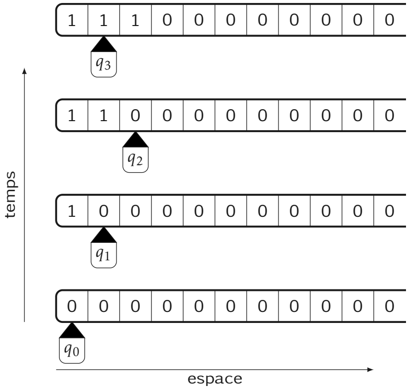


$$(q,a) \longrightarrow (q',a',\uparrow)$$

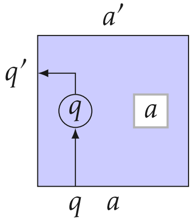


Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.

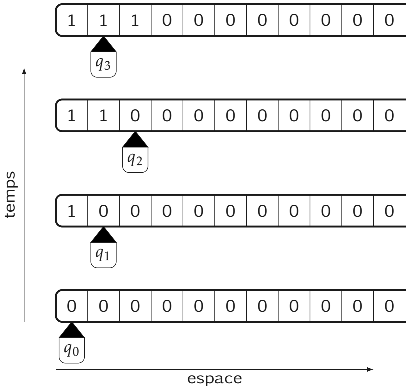


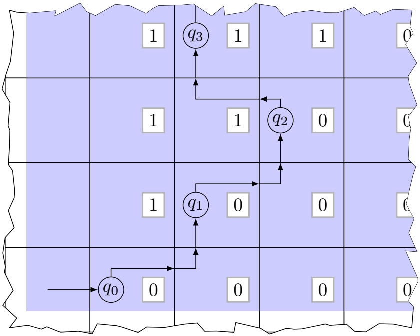
$$(q, a) \longrightarrow (q', a', \leftarrow)$$



Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_\mathcal{F}$ is empty, given $\mathcal{F}$ as input.

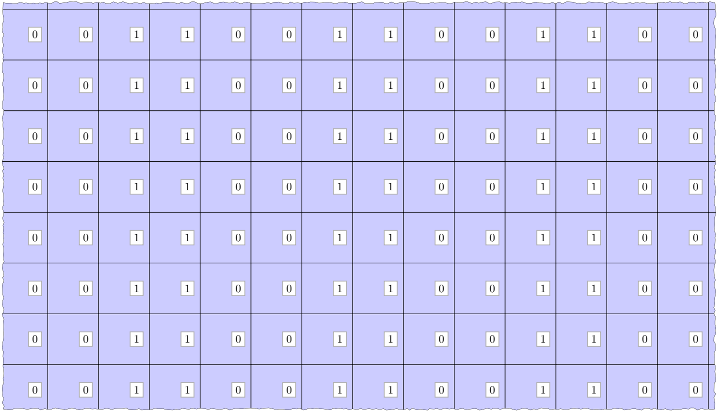




Infinite tiling ⇔ Turing machine **does not halt**

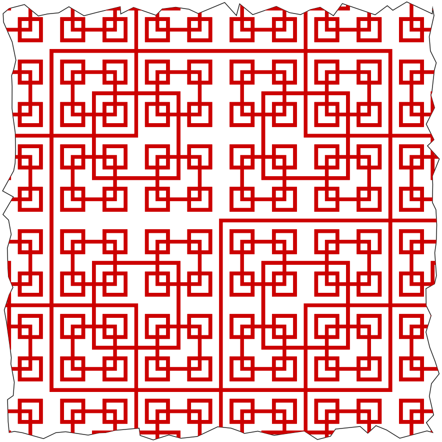
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_\mathcal{F}$ is empty, given $\mathcal{F}$ as input.



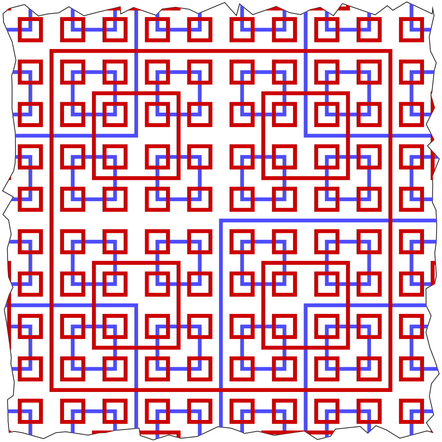
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



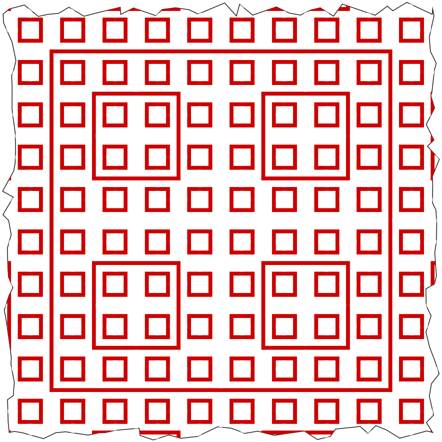
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



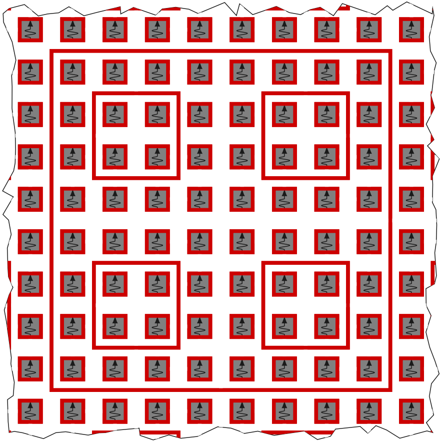
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



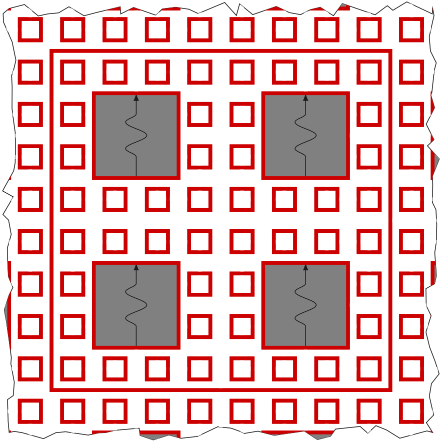
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_\mathcal{F}$ is empty, given $\mathcal{F}$ as input.



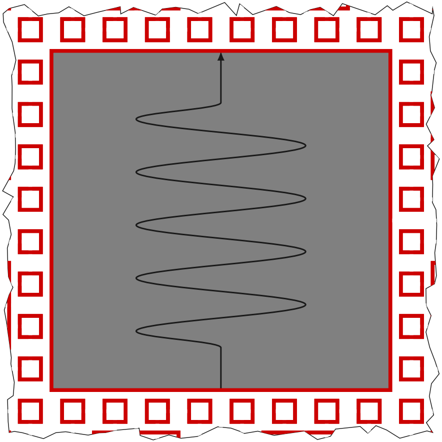
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is undecidable to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



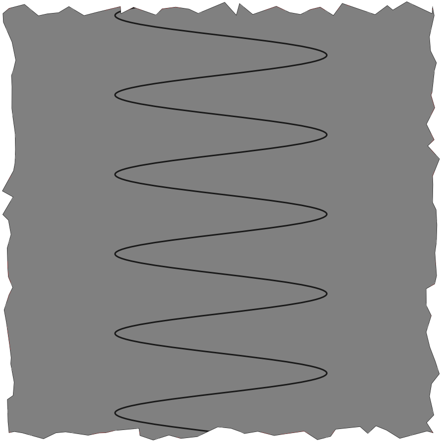
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



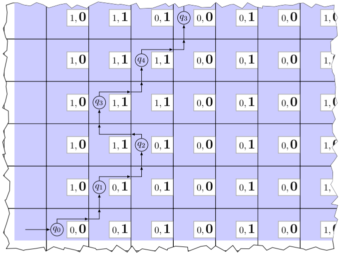
Nothing is easy in $d \geq 2$

Theorem [Berger 1964] It is **undecidable** to know whether $\mathcal{X}_{\mathcal{F}}$ is empty, given $\mathcal{F}$ as input.



SFTs without any computable configuration

There exists a TM M that **does not halt** only on **non computable oracles**:



The quarter plane may be tiled iff M does not halt on x.

Theorem [Hanf-Myers 1974] **There exist SFTs containing only non computable points.**

The right tool

SFTs are **dynamical systems**, some quantities/concepts are important:

- Topological Entropy : measure of the growth of the number of patterns
- Number of periodic points
- Subactions, non-expansive directions, growth-type invariants...

The right tool

SFTs are **dynamical systems**, some quantities/concepts are important:

- Topological Entropy : measure of the growth of the number of patterns

  [Hochman & Meyerovitch 2010] **Entropies** of SFTs correspond to the **upper semi-computable** real numbers.

- Number of periodic points

  The functions counting the number of periodic points are exactly the functions of **#P**.

- Subactions, non-expansive directions, growth-type invariants...

Turing degrees

- $x \leq_T y$ if there exists a **TM that outputs x with input y**.
- $x \equiv_T y$ if $x \leq_T y$ and $x \geq_T y$.
- A **Turing degree** is an equivalence class for $\equiv_T$. The degree of x is noted $\deg_T x$.

The simplest degree is $\mathbf{0}$: the degree of computable objects.

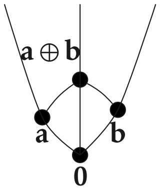
- Turing degree of a configuration.
- **Turing degree spectrum** of a subshift:

$$\mathbf{Sp}(X) = \left\{\deg_T x \;\middle|\; x \in X\right\}$$

Turing degrees

There exists a degree $\mathbf{a} \oplus \mathbf{b}$ which is the smallest above both $\mathbf{a}$ and $\mathbf{b}$.

- Every Turing degree contains exactly $\aleph_0$ elements.
- There are $2^{\aleph_0}$ Turing degrees.
- There are at most $\aleph_0$ degrees below any degree.
- There are $2^{\aleph_0}$ degrees above each degree.



$\mathbf{0}$ the degree of computable sequences.

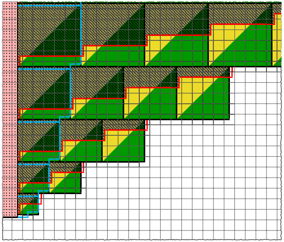
There exist **incomparable degrees** $\mathbf{a}, \mathbf{b}$:

$$\mathbf{a} \not\leq_T \mathbf{b} \text{ and } \mathbf{b} \not\leq_T \mathbf{a}$$

Turing degree spectra of subshifts

Theorem For any **effectively closed** set of Turing degrees S, there exists an SFT X with the **same spectrum up to 0**:

$$\mathbf{Sp}(S) \cup \{\mathbf{0}\} = \mathbf{Sp}(X)$$



Turing degree spectra of subshifts

Theorem For any **closed** set of Turing degrees S, there exists an **subshift** X with the **same spectrum up to 0**:

$$\mathbf{Sp}(S) \cup \{\mathbf{0}\} = \mathbf{Sp}(X)$$

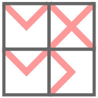
$$\cdots - - - - - - - - - - 1 - 0 - - 1 - - - 1 - - - - 0 - - - - - - 1 - - \cdots$$

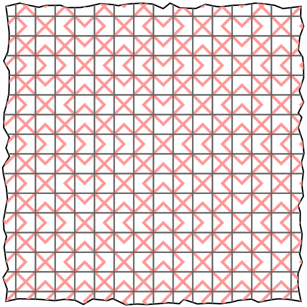
Minimality

Definition A subshift X is **minimal** iff **all** its **configurations contain the same patterns**.

Uniform recurrence. For every pattern, there exists a window in which it will always appear.

Example :





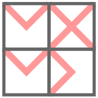
Theorem **Every subshift contains a minimal subshift**

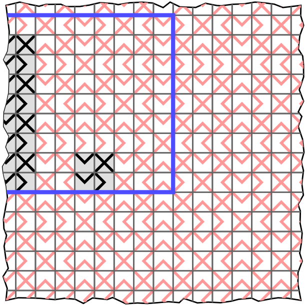
Minimality

Definition A subshift X is minimal iff all its configurations contain the same patterns.

Uniform recurrence. For every pattern, there exists a window in which it will always appear.

Example :





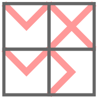
Theorem Every subshift contains a minimal subshift

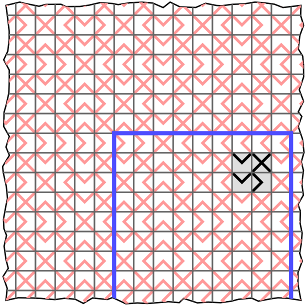
Minimality

Definition A subshift X is minimal iff all its configurations contain the same patterns.

Uniform recurrence. For every pattern, there exists a window in which it will always appear.

Example :





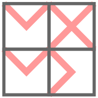
Theorem Every subshift contains a minimal subshift

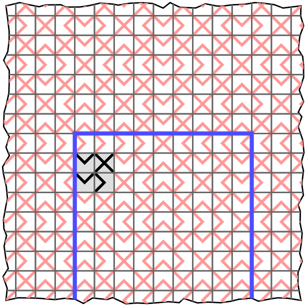
Minimality

Definition A subshift X is minimal iff all its configurations contain the same patterns.

Uniform recurrence. For every pattern, there exists a window in which it will always appear.

Example :





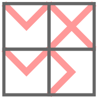
Theorem Every subshift contains a minimal subshift

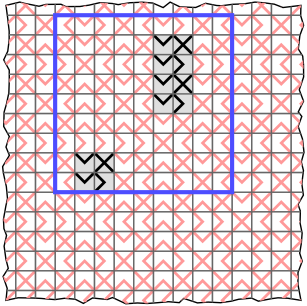
Minimality

Definition A subshift X is minimal iff all its configurations contain the same patterns.

Uniform recurrence. For every pattern, there exists a window in which it will always appear.

Example :

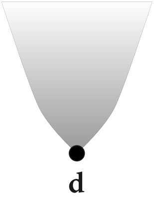




Theorem Every subshift contains a minimal subshift

Minimality and Turing degrees

Theorem Let X be a non finite **minimal subshift**, then $\mathbf{Sp}(X)$ contains the **cone** of degrees **above any of its points**.



Cone above $\mathbf{d}$:

$$\mathcal{C}_{\mathbf{d}} = \left\{\mathbf{d}' \,\middle|\, \mathbf{d}' \geq_T \mathbf{d}\right\}$$

Spectra of minimal SFTs

Theorem Let X be a subshift, and $x \in X$ be an aperiodic recurrent point, then $\mathbf{Sp}(X)$ contains the cone above $\deg_T x$.

Proof. We build two **computable** functions:

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

such that $(x, y) \in A \times \{0,1\}^{\mathbb{N}}$:

$$dec(enc(x, y)) = y$$

Spectra of minimal SFTs

Theorem Let X be a subshift, and $x \in X$ be an aperiodic recurrent point, then $\mathbf{Sp}(X)$ contains the cone above $\deg_T x$.

Proof. We build two **computable** functions:

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$ $\qquad\qquad$ $\deg_T(enc(x,y)) \leq_T \deg_T x \oplus y$
- $dec : A \to \{0,1\}^{\mathbb{N}}$ $\qquad\qquad$ $\deg_T(dec(x)) \leq_T \deg_T x$

such that $(x,y) \in A \times \{0,1\}^{\mathbb{N}}$:

$$dec(enc(x,y)) = y$$

Spectra of minimal SFTs

Theorem Let X be a subshift, and $x \in X$ be an aperiodic recurrent point, then $\mathbf{Sp}(X)$ contains the cone above $\deg_T x$.

Proof. We build two **computable** functions:

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$ $\qquad \deg_T(enc(x,y)) \leq_T \deg_T x \oplus y$
- $dec : A \to \{0,1\}^{\mathbb{N}}$ $\qquad \deg_T(dec(x)) \leq_T \deg_T x$

such that $(x,y) \in A \times \{0,1\}^{\mathbb{N}}$:

$$dec(enc(x,y)) = y$$

So we have this inequality:

$$\deg_T(y) \leq_T \deg_T(enc(x,y)) \leq_T \deg_T(\sup(x,y))$$

In particular if we **choose y such that $\deg_T(y) \geq \deg_T(x)$**, then

$$\deg_T(enc(x,y)) = \deg_T(y)$$

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

By induction: from a word c_i construct c_{i+1}.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

Minimality: we know that c_i appears in any window of sufficiently big.

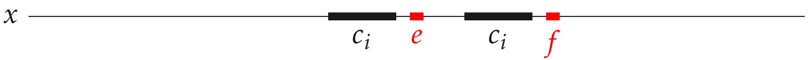
Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

Minimality: we know that c_i appears in any window of sufficiently big.

Idea of the proof : in dimension 1

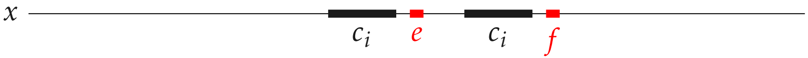
- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$



x cannot be periodic since X is non finite.

Idea of the proof : in dimension 1

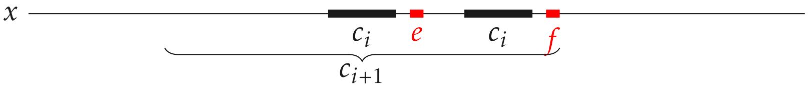
- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$



x cannot be periodic since X is non finite.
$e < f$ or $e > f$, both cases will appear somewhere.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$



c_{i+1} is constructed according to y_i:

- if $y_i = 0$, take $e < f$,
- if $y_i = 1$, take $e > f$.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

Start with $c_0 = x_0$, and iterate.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

Start with $c_0 = x_0$, and iterate.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

Start with $c_0 = x_0$, and iterate.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

Start with $c_0 = x_0$, and iterate.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$

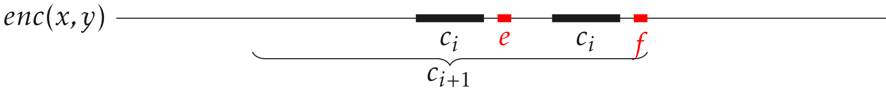
x

$$\lim_{\infty} c_i = enc(x, y)$$

Start with $c_0 = x_0$, and iterate.

Idea of the proof : in dimension 1

- $enc : A \times \{0,1\}^{\mathbb{N}} \to A$
- $dec : A \to \{0,1\}^{\mathbb{N}}$



Start with $c_0 = x_0$, and look for the first differing letters e, f.

- if $e > f$ then $y_i = 1$
- if $e < f$ then $y_i = 0$

We now know c_1 and can look for c_2 and so on...

□

Complexity function

Dimension 1 from now on.

Most results do not translate to higher dimensions.

Definition The **complexity function**:

$$c_n(X)$$

counts the **number of patterns of size n**.

Linear complexity

The trivial cases

- $c_n(X) < n + 1 \Rightarrow$ Only periodic configurations

$$\cdots 123123123123123 \cdots$$

- $c_n(X) = n + k$ and eventually periodic on both sides

$$\cdots 000000100000000 \cdots$$

$$\mathbf{Sp}(X) = \mathbf{0}$$

Linear complexity

Sturmian subshifts

- Low complexity : $c_n(X) = n+1$
- No periodic points
- Only aperiodic recurrent points

$$\cdots 101001\underbrace{001010}_{w}010\underbrace{100100}_{w'}1010\cdots$$

- If w, w' have the same length then $||w|_1 - |w'|_1| \leq 1$.
- Density of 1s tends to $\{\alpha\}$.

$$\mathbf{Sp}(X) = \mathcal{C}_{\deg_T \alpha}$$

Linear complexity

Theorem If $c_n(X) \sim tn$ then, $\mathbf{Sp}(X)$ contains **at most k isolated degrees and k cones with $k + k' \leq t$**.

Lemma If $c_n X \sim tn$ then X contains **at most t non recurrent aperiodic configurations**.

at most t isolated degrees.

Lemma If $c_n X \sim tn$ and X contains k **non recurrent aperiodic** configurations then X contains **at most $t - k$ recurrent aperiodic configurations** with different language.

$$\{\mathcal{L}(x) \mid x \text{ aperiodic recurrent}\}$$

at most $t - k$ cones.

not directly though...

Linear complexity

Aperiodic recurrent configurations

Lemma If x is aperiodic recurrent with linear complexity, then

$$x \geq_T \mathcal{L}(x)$$

Theorem [Cassaigne 1995] If x has linear growth, then $c_{n+1}(x) - c_n(x)$ is bounded by a constant.

There exists N and M such that for **infinitely many $n > N$**:

$$c_{n+1}(X) - c_n(X) = M$$

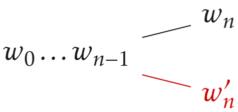
Linear complexity

Aperiodic recurrent configurations

There exists N and M such that for **infinitely many $n > N$**:

$$c_{n+1}(X) - c_n(X) = M$$

Some words can be followed by different letters:



There are exactly M **choices** for all words of length n.

Linear complexity

Aperiodic recurrent configurations

Lemma If x is aperiodic recurrent with linear complexity, then

$$x \geq_T \mathcal{L}(x)$$

Take x as an oracle and output $\mathcal{L}(x)$:

- hardcode N, M
- scan x and find all words of the same length with several choices : S

Linear complexity

Aperiodic recurrent configurations

Lemma If x is aperiodic recurrent with linear complexity, then

$$x \geq_T \mathcal{L}(x)$$

Take x as an oracle and output $\mathcal{L}(x)$:

- hardcode N, M
- scan x and find all words of the same length with several choices : S
- Find all n-letter words:

- We now have $\mathcal{L}_k(x)$ for $k \leq n$.

Linear complexity

Aperiodic recurrent configurations

Lemma If x is aperiodic recurrent with linear complexity, then

$$x \geq_T \mathcal{L}(x)$$

Take x as an oracle and output $\mathcal{L}(x)$:

- hardcode N, M
- scan x and find all words of the same length with several choices : S
- Find all n-letter words:

- We now have $\mathcal{L}_k(x)$ for $k \leq n$.

Linear complexity

Aperiodic recurrent configurations

Lemma If x is aperiodic recurrent with linear complexity, then

$$x \geq_T \mathcal{L}(x)$$

Take x as an oracle and output $\mathcal{L}(x)$:

- hardcode N, M
- scan x and find all words of the same length with several choices : S
- Find all n-letter words:

- We now have $\mathcal{L}_k(x)$ for $k \leq n$.

Linear complexity

Last ingredient:

Lemma If x is aperiodic recurrent, there exists y such that

$$\mathcal{L}(x) = \mathcal{L}(y) \text{ and } \deg_T y = \deg_T \mathcal{L}(x)$$

Theorem If $c_n(X) \sim tn$ then, $\mathbf{Sp}(X)$ contains **at most k isolated degrees and k cones with $k + k' \leq t$**.

Linear complexity

Theorem There exist linear complexity subshifts with k cones and k' isolated degrees for any k, k'.

- k cones: union of Sturmians
- k' isolated degrees:

$$\cdots - - - - - - - - s_0 \underbrace{- - \cdots - -}_{f(0)} s_1 \underbrace{- - \cdots - -}_{f(1)} s_2 \underbrace{- - \cdots - -}_{f(2)} s_3 - - - - \cdots$$

  - $s \in \{0, 1\}^{\mathbb{N}}$
  - f computable
  - same degree as s
  - linear growth

Exponential complexity: positive entropy

Exponential complexity (=positive entropy):

$$c_n(X) \sim a^n$$

Theorem If $h(X) > 0$, then $\mathbf{Sp}(X)$ contains a cone.

Theorem Any spectrum containing a cone can be realized by a subshift with entropy in this cone.

The inbetweeners

- Constant Only $\mathbf{0}$.
- Linear Finite number of cones and isolated degrees.
- Exponential Contain a cone.

The inbetweeners

- Constant Only $\mathbf{0}$.
- Linear Finite number of cones and isolated degrees.
- Exponential Contain a cone.
- Superlinear ?

Slow superlinear complexity

Theorem For any countable set of degrees, $S = \{\mathbf{d_1}, \mathbf{d_2}, \dots\}$ there exists subshifts with **arbitrarily slow superlinear complexity** and **spectrum** $\bigcup \mathcal{C}_{\mathbf{d_i}}$.

Proof idea.

Take some increasing unbounded f.

Take $(\alpha_k)_{k\in\mathbb{N}}$ and $(m_k)_{k\in\mathbb{N}}$ such that

$$\alpha_k \to \alpha_0 \quad \text{and} \quad \mathcal{L}_{m_k}\left(S_{\alpha_k}\right) = \mathcal{L}_{m_k}\left(S_{\alpha_0}\right) \quad \text{and} \quad m_{f(n)/2} > n$$

Define

$$X = \bigcup_k S_{\alpha_k} \qquad \text{its spectrum is } \mathbf{Sp}(X) = \bigcup \mathbf{d} \in S\mathcal{C}_{\mathbf{d}}$$

it is **closed** since $\alpha_k \to \alpha_0$, and hence a subshift.

$c_n X$ **is bounded by** $nf(n)$**.**

Slow superlinear complexity

Theorem For any countable set of degrees, $S = \{\mathbf{d_1}, \mathbf{d_2}, \dots\}$ there exists subshifts with **arbitrarily slow superlinear complexity** and **spectrum** $S \cup \{\mathbf{0}\}$.

Proof idea.
For each degree $\mathbf{d_i} \in S$ include s of degree $\mathbf{d_i}$:

$$\cdots 0000.10^{2^1}10^{2^2}10^{2^3}1\cdots 10^{2^{m_i}}10^{2^{m_i}+1+s_1}10^{2^{m_i}+2+s_2}1\cdots$$

Limit points:

- $\cdots 000010000 \cdots$
- $\cdots 000000000 \cdots$
- $\cdots 000010^{2^1}1\cdots 10^{2^k}1\cdots$

□

The inbetweeneers

- Constant Only $\mathbf{0}$.
- Linear Finite number of cones and isolated degrees.
- Exponential Contains a cone.
- Superlinear $\sim$ Anything is possible. Tradeoff
  - Countable unions, any superlinear growth
  - Unions, any superlinear computable growth
- Subexponential $\sim$ Anything is possible
- The rest $\sim$ Anything is possible