

Tilings and Cellular Automata, part I.

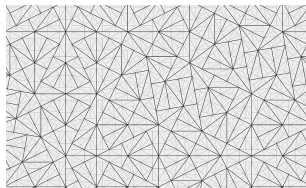
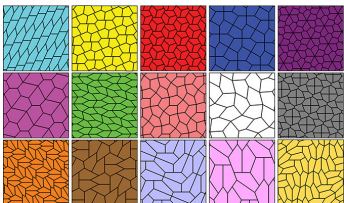
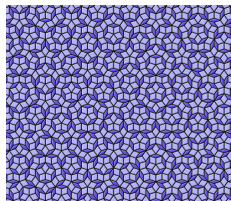
EJCIM 2017, LIP (ENS de Lyon)

Nathalie Aubrun, Guillaume Theyssier

24th January 2017

Tilings?

"Cover the plane with (arbitrarily many) copies of some basic tiles."



Outline of the talk.

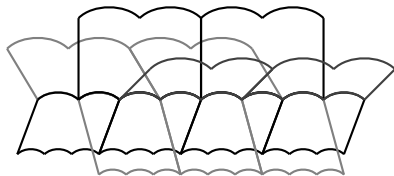
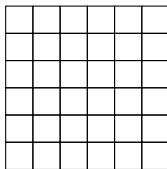
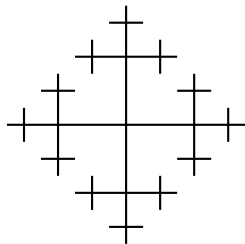
- 1 Introduction: Symbolic Dynamics and tilings
- 2 Dynamical systems?
- 3 Dimension 2 ($G = \mathbb{Z}^2$)
- 4 Undecidability of DP on \mathbb{Z}^2 , proof I
- 5 Undecidability of DP on \mathbb{Z}^2 , proof II

Configurations, patterns and cylinders

- ▶ Let A be a finite alphabet, G be a finitely generated group.
- ▶ Colorings $x : G \rightarrow A$ are called **configurations**.
- ▶ A **pattern** is a finite configuration $p : S \rightarrow A$.

Examples:

$$A = \{ \text{green square}, \text{orange square} \}$$

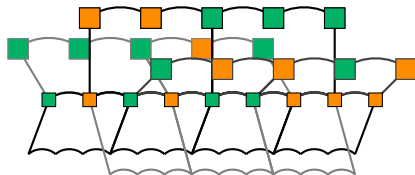
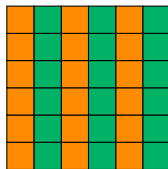
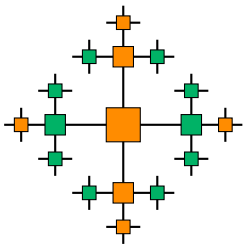


Configurations, patterns and cylinders

- ▶ Let A be a finite alphabet, G be a finitely generated group.
- ▶ Colorings $x : G \rightarrow A$ are called **configurations**.
- ▶ A **pattern** is a finite configuration $p : S \rightarrow A$.

Examples:

$$A = \{ \blacksquare, \blacksquare \}$$



The Cantor space A^G ?

- ▶ Endowed with the pro-discrete topology A^G is a **compact** and **metrizable** set.
- ▶ **Cylinders** form a clopen basis

$$[a]_g = \{x \in A^G \mid x_g = a\}.$$

- ▶ A **pattern** is a finite intersection of cylinders, or equivalently a finite configuration $p : S \rightarrow A$
- ▶ A **metric** for the cylinder topology is

$$d(x, y) = 2^{-\inf\{|g| \mid g \in G: x_g \neq y_g\}},$$

where $|g|$ is the length of the shortest path from 1_G to g in $\Gamma(G, S)$.

Subshifts: topological definition

The **shift** σ is the natural action of G on A^G by translation:

$$\sigma^g(x)_h = x_{g^{-1} \cdot h} \text{ for all } x \in A^G.$$

Subshifts: topological definition

The **shift** σ is the natural action of G on A^G by translation:

$$\sigma^g(x)_h = x_{g^{-1} \cdot h} \text{ for all } x \in A^G.$$

(Topological) Definition: subshift

A **subshift** is a closed and σ -invariant subset of A^G .

Examples:

- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{no two consecutive 1's in } x\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid x_i = 0 \Rightarrow i \text{ is even}\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{finite blocks of 1's are of even length}\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{finite blocks of 1's are of prime length}\}$?
- ▶ $X = \{x \in \{0, 1\}^G \mid |\{g \in G : x_g = 1\}| = 1\}$?
- ▶ $X = \{x \in \{0, 1\}^G \mid |\{g \in G : x_g = 1\}| \leq 1\}$?

Subshifts: combinatorial definition

(Combinatorial) Definition: subshift

Let F be a set of finite patterns. The **subshift** defined by the set of forbidden patterns F is the set

$$X_F = \{x \in A^G, \text{no pattern of } F \text{ appears in } x\}.$$

Subshifts: combinatorial definition

(Combinatorial) Definition: subshift

Let F be a set of finite patterns. The **subshift** defined by the set of forbidden patterns F is the set

$$X_F = \{x \in A^G, \text{no pattern of } F \text{ appears in } x\}.$$

Proposition

The topological and combinatorial definitions coincide.

Compactness of A^G

Proposition

The Cantor space A^G is compact for the pro-discrete topology.

Concretely:

- ▶ X_F is non-empty iff there exist *arbitrarily big* finite patterns that avoid F (for instance every ball B_n can be colored avoiding F).
- ▶ If there exist *arbitrarily big* finite patterns with no letter a that avoid F , there exists a configuration in X_F with no letter a .

Subshifts: at the interplay between several domains

- ▶ Dynamical Systems (Symbolic Dynamics)
- ▶ Tilings theory (by Wang tiles)
- ▶ Cellular automata (see next course by G. Theyssier)

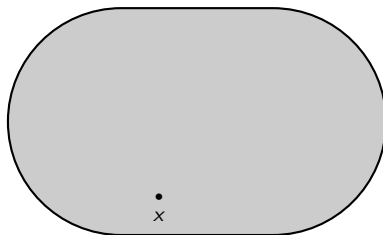
Outline of the talk.

- 1 Introduction: Symbolic Dynamics and tilings
- 2 Dynamical systems?
- 3 Dimension 2 ($G = \mathbb{Z}^2$)
- 4 Undecidability of DP on \mathbb{Z}^2 , proof I
- 5 Undecidability of DP on \mathbb{Z}^2 , proof II

Discrete dynamical systems

(X, F) is a **discrete dynamical system** if:

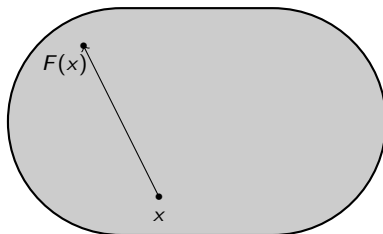
- ▶ X is a topological compact space, called the **phase space**
- ▶ F is a continuous map : $X \rightarrow X$



Discrete dynamical systems

(X, F) is a **discrete dynamical system** if:

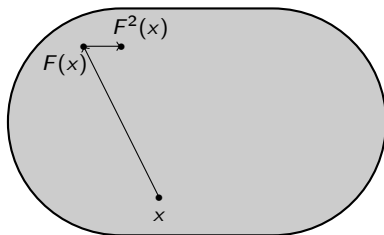
- ▶ X is a topological compact space, called the **phase space**
- ▶ F is a continuous map : $X \rightarrow X$



Discrete dynamical systems

(X, F) is a **discrete dynamical system** if:

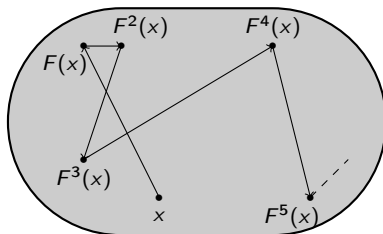
- ▶ X is a topological compact space, called the **phase space**
- ▶ F is a continuous map : $X \rightarrow X$



Discrete dynamical systems

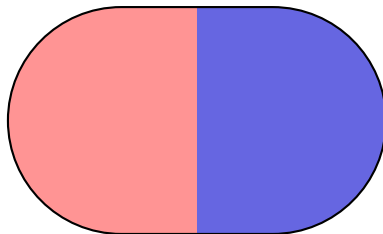
(X, F) is a **discrete dynamical system** if:

- ▶ X is a topological compact space, called the **phase space**
- ▶ F is a continuous map : $X \rightarrow X$



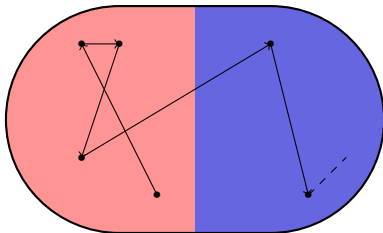
Coding of the orbits

- ▶ $X = \bigcup_{i=1}^n X_i$ a partition of the phase space X
- ▶ a color a_i associated with each X_i



Coding of the orbits

- ▶ $X = \bigcup_{i=1}^n X_i$ a partition of the phase space X
- ▶ a color a_i associated with each X_i
- ▶ orbit $(F^n(x))_{n \in \mathbb{N}}$ coded by a sequence $y \in \{a_1, \dots, a_n\}^{\mathbb{N}}$



Dynamical Systems and subshifts

- ▶ If the dynamical system (X, F) is *invertible* and *expansive*, then $X = \bigcup_{i=1}^n X_i$ can be chosen so that the set of coding of orbits

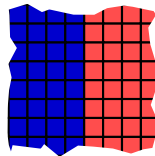
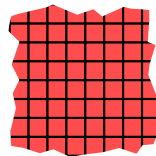
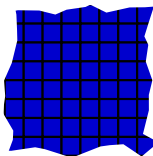
$$\tilde{X} = \{y \in \{a_1, \dots, a_n\}^{\mathbb{Z}} \mid \exists x \in X, \forall k \in \mathbb{Z}, F^k(x) \in X_i\}$$

is a subshift in **one-to-one correspondence** with (X, F) .

- ▶ Dynamical properties of the original system (X, F) can be read on the corresponding subshift (\tilde{X}, σ) .
- ▶ If the partition $X = \bigcup_{i=1}^n X_i$ is well-chosen (Markov partition), the subshift \tilde{X} is an SFT!

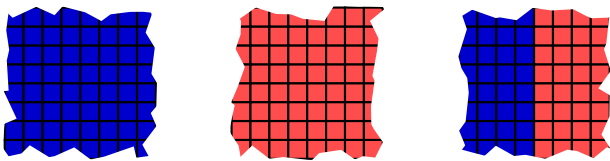
Classes of subshifts: Subshifts of finite type (SFT)

- ▶ Sets of configurations that avoid a **finite** set of forbidden patterns.
- ▶ For instance $X_{\left\{ \begin{array}{c} \text{red} \text{ blue} \\ \text{blue} \end{array} \right\}, \left\{ \begin{array}{c} \text{blue} \\ \text{red} \end{array} \right\}, \left\{ \begin{array}{c} \text{red} \\ \text{blue} \end{array} \right\} \right\}$ contains the following configurations (in 2D):



Classes of subshifts: Subshifts of finite type (SFT)

- ▶ Sets of configurations that avoid a **finite** set of forbidden patterns.
- ▶ For instance $X_{\{ \begin{smallmatrix} \text{red} & \text{blue} \\ \text{blue} & \text{red} \end{smallmatrix}, \begin{smallmatrix} \text{blue} \\ \text{red} \end{smallmatrix}, \begin{smallmatrix} \text{red} \\ \text{blue} \end{smallmatrix} \}}$ contains the following configurations (in 2D):



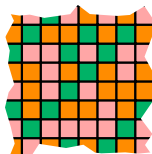
Definition: subshift of finite type (SFT)
A **subshift of finite type (SFT)** is a subshift that can be defined by a finite set of forbidden patterns.

- ▶ simplest class with respect to the combinatorial definition
- ▶ 2D-SFT \equiv Wang tilings

Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



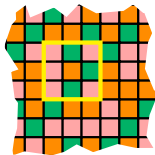
$$\Phi(x) \in B^{\mathbb{Z}^2}$$



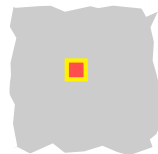
Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



$$\Phi(x) \in B^{\mathbb{Z}^2}$$



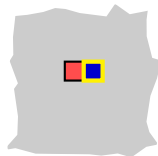
Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



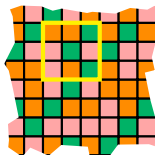
$$\Phi(x) \in B^{\mathbb{Z}^2}$$



Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



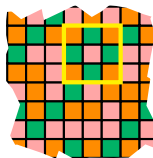
$$\Phi(x) \in B^{\mathbb{Z}^2}$$



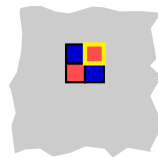
Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



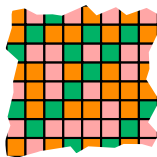
$$\Phi(x) \in B^{\mathbb{Z}^2}$$



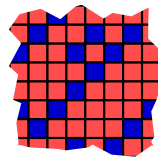
Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



$$\Phi(x) \in B^{\mathbb{Z}^2}$$



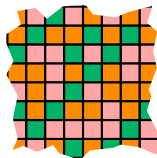
Definition: sofic subshift

A **sofic subshift** is the factor of an SFT.

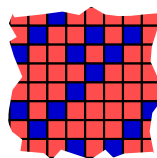
Classes of subshifts: Sofic subshifts

A factor map $\Phi : A^G \rightarrow B^G$ is given by a local map ϕ (or equivalently Φ is a continuous and σ -commuting map):

$$x \in A^{\mathbb{Z}^2}$$



$$\Phi(x) \in B^{\mathbb{Z}^2}$$



Definition: sofic subshift

A **sofic subshift** is the factor of an SFT.

- ▶ Recodings of SFT, with local neighborhood.
- ▶ In 1D ($G = \mathbb{Z}$), sofic subshifts are exactly those recognized by finite automata.

Examples

Which subshifts are SFTs? sofic subshifts?

Examples:

- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{no two consecutive 1's in } x\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{finite blocks of 1's are of even length}\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{finite blocks of 1's are of prime length}\}$?
- ▶ $X = \{x \in \{0, 1\}^G \mid |\{g \in G : x_g = 1\}| \leq 1\}$?

Big questions in Symbolic Dynamics

- ▶ Classify SFTs/sofic subshifts up to conjugacy.
- ▶ Find conjugacy invariants.
- ▶ Find factors of SFTs with equal entropy?
- ▶ Decide dynamical properties? (injectivity, surjectivity, expansiveness. . .)

Dimension 1

Given a subshift $X \subset A^{\mathbb{Z}}$, one can consider its **language** $\mathcal{L}(X)$ defined by

$$\mathcal{L}(X) = \bigcup_{n \in \mathbb{N}} \mathcal{L}_n(X)$$

where

$$\mathcal{L}_n(X) = \{w \in A^n \mid \exists x \in X, x_i = w_i \forall i = 1 \dots n\}.$$

Remark: We have $X = \overline{X_{\mathcal{L}(X)}}$, and $\overline{\mathcal{L}(X)}$ is the biggest (for inclusion) set of patterns that defines X .

Graph representation of SFTs and sofic subshifts

Proposition

A subshift X is sofic iff its language $\mathcal{L}(X)$ is rational.

Graph representation of SFTs and sofic subshifts

Proposition

A subshift X is sofic iff its language $\mathcal{L}(X)$ is rational.

Proposition

A subshift X is sofic iff it is the set of labels of bi-infinite paths of a finite edge-labeled graph.

Examples:

- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{no two consecutive 1's in } x\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid \text{finite blocks of 1's are of even length}\}$?
- ▶ $X = \{x \in \{0, 1\}^{\mathbb{Z}} \mid |\{i \in \mathbb{Z} : x_i = 1\}| \leq 1\}$?

What can be read on the graph?

- ▶ Existence of a configuration in X_F .
- ▶ Periodic configurations.
- ▶ Compute the entropy from the graph matrix.
- ▶ etc. . .

Outline of the talk.

- 1 Introduction: Symbolic Dynamics and tilings
- 2 Dynamical systems?
- 3 Dimension 2 ($G = \mathbb{Z}^2$)**
- 4 Undecidability of DP on \mathbb{Z}^2 , proof I
- 5 Undecidability of DP on \mathbb{Z}^2 , proof II

SFTs and Wang tiles

We now fix $G = \mathbb{Z}^2$.

Wang tiles



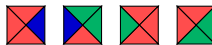
Neighborhood rule



SFTs and Wang tiles

We now fix $G = \mathbb{Z}^2$.

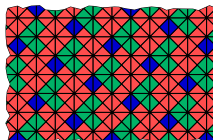
Wang tiles



Neighborhood rule



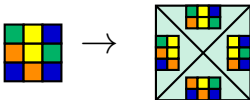
X_τ set of valid tilings by τ



The Domino problem

SFT $\approx X_\tau$

- ▶ Tilings by Wang tiles can be seen as an SFT.
- ▶ Every SFT can be encoded inside a finite set of Wang tiles



Domino problem on \mathbb{Z}^2

Input: A finite set of Wang tiles τ .

Output: **Yes** if there exists a valid tiling by τ , **No** otherwise.

Periodicity

We can define two notions of **periodic** configuration:

- ▶ A configuration $x \in A^{\mathbb{Z}^2}$ is **weakly periodic** if x admits a non-trivial direction \vec{u} of periodicity.
- ▶ A configuration $x \in A^{\mathbb{Z}^2}$ is **strongly periodic** if x admits two non-collinear directions \vec{u}, \vec{v} of periodicity.

Periodicity

We can define two notions of **periodic** configuration:

- ▶ A configuration $x \in A^{\mathbb{Z}^2}$ is **weakly periodic** if x admits a non-trivial direction \vec{u} of periodicity.
- ▶ A configuration $x \in A^{\mathbb{Z}^2}$ is **strongly periodic** if x admits two non-collinear directions \vec{u}, \vec{v} of periodicity.

Proposition

On \mathbb{Z}^2 , if an SFT contains a weakly periodic configuration, then it contains a strongly periodic one.

Proof: on the blackboard.

Domino problem and periodicity on \mathbb{Z}^2

Wang's conjecture (1961)

If a set of Wang tiles can tile the plane, then they can always be arranged to do so periodically.

Domino problem and periodicity on \mathbb{Z}^2

Wang's conjecture (1961)

A non-empty SFT contains a periodic configuration.

Domino problem and periodicity on \mathbb{Z}^2

Wang's conjecture (1961)

A non-empty SFT contains a periodic configuration.

Suppose Wang's conjecture is true. Then you can decide **DP** !

Semi-algorithm 1:

- 1 gives a finite periodic pattern, if it exists
- 2 loops otherwise

Semi-algorithm 2:

- 1 gives an integer n so that there is no $[1; n] \times [1; n]$ locally admissible pattern, if it exists
- 2 loops otherwise

Domino problem and periodicity on \mathbb{Z}^2

Wang's conjecture (1961)

A non-empty SFT contains a periodic configuration.

Suppose Wang's conjecture is true. Then you can decide **DP** !

Semi-algorithm 1:

- 1 gives a finite periodic pattern, if it exists
- 2 loops otherwise

Semi-algorithm 2:

- 1 gives an integer n so that there is no $[1; n] \times [1; n]$ locally admissible pattern, if it exists
- 2 loops otherwise

Consequence

The undecidability of **DP** implies existence of an aperiodic SFT.

Outline of the talk.

- 1 Introduction: Symbolic Dynamics and tilings
- 2 Dynamical systems?
- 3 Dimension 2 ($G = \mathbb{Z}^2$)
- 4 Undecidability of DP on \mathbb{Z}^2 , proof I**
- 5 Undecidability of DP on \mathbb{Z}^2 , proof II

Attempt to prove undecidability of DP

Idea: encode **Turing machines** inside Wang tiles.

- ▶ Undecidability of the Halting problem of Turing machines.
- ▶ Reduction from the Halting problem of Turing machines.

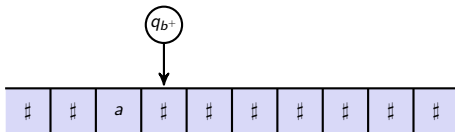
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



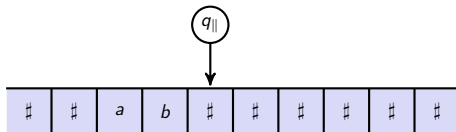
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



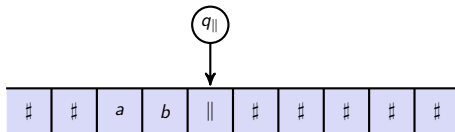
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



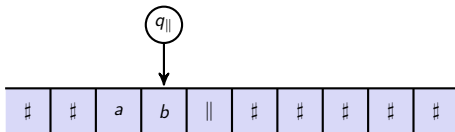
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



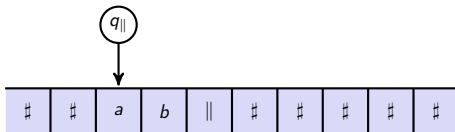
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



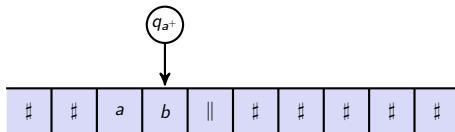
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



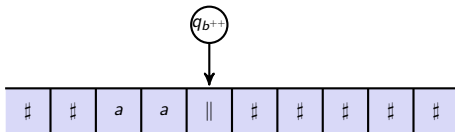
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



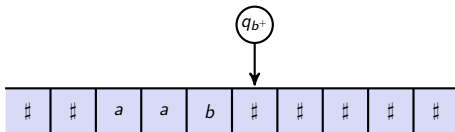
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



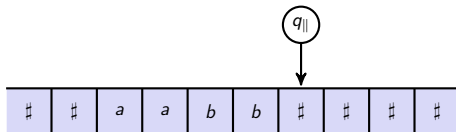
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



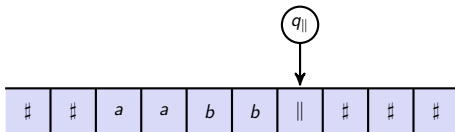
Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$



Turing machines

$\delta(q, x)$		Symbol x			
		a	b	\parallel	$\#$
State q	q_0	\perp	\perp	\perp	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	\perp	$(q_{b^{++}}, a, \rightarrow)$	\perp	\perp
	q_{b^+}	\perp	\perp	\perp	$(q_{\parallel}, b, \rightarrow)$
	$q_{b^{++}}$	\perp	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	\perp
	q_{\parallel}	$(q_{a^+}, a, \rightarrow)$	$(q_{\parallel}, b, \leftarrow)$	$(q_{\parallel}, \parallel, \leftarrow)$	$(q_{\parallel}, \parallel, \cdot)$

Theorem (Turing, 1936)

The Halting problem (to know whether a Turing machine \mathcal{M} halts on input w or not) is undecidable.

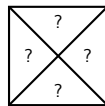
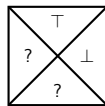
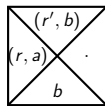
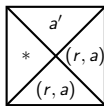
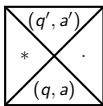
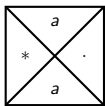
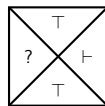
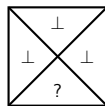
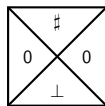
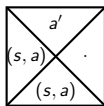
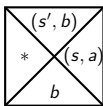
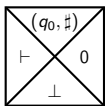
Theorem

The Blank tape Halting problem (to know whether a Turing machine \mathcal{M} halts on the empty input) is undecidable.

Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

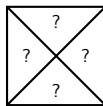
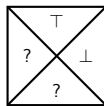
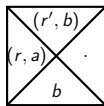
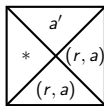
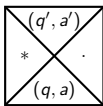
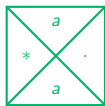
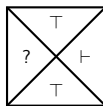
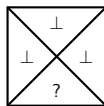
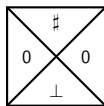
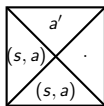
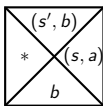
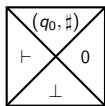
- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

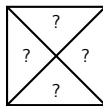
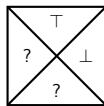
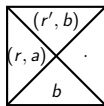
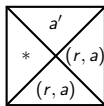
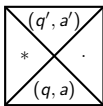
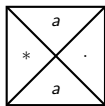
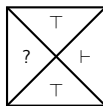
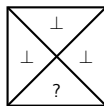
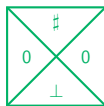
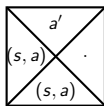
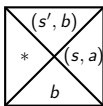
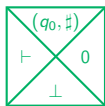
- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

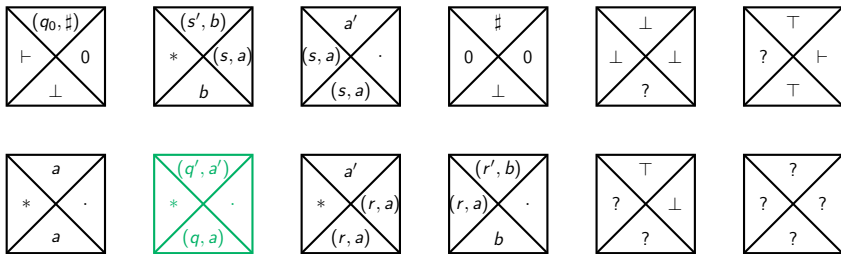
- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

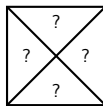
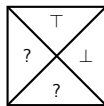
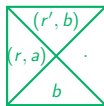
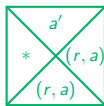
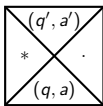
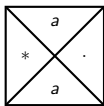
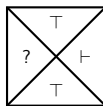
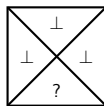
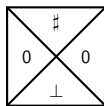
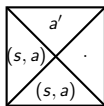
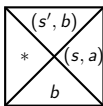
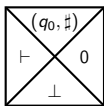
- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

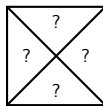
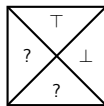
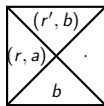
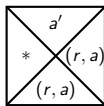
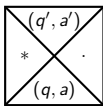
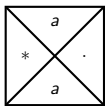
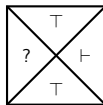
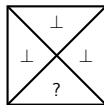
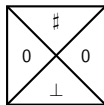
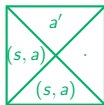
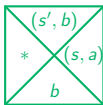
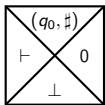
- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

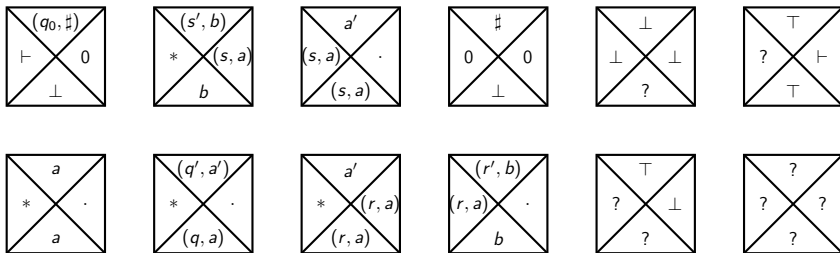
- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

- ▶ no computation head
- ▶ initial configuration $(\infty \# \infty, q_0)$
- ▶ $\delta(q, a) = (q', a', \cdot)$
- ▶ $\delta(r, a) = (r', a', \rightarrow)$
- ▶ $\delta(s, a) = (s', a', \leftarrow)$



We want: τ admits a tiling iff \mathcal{M} does not halt on the empty input.

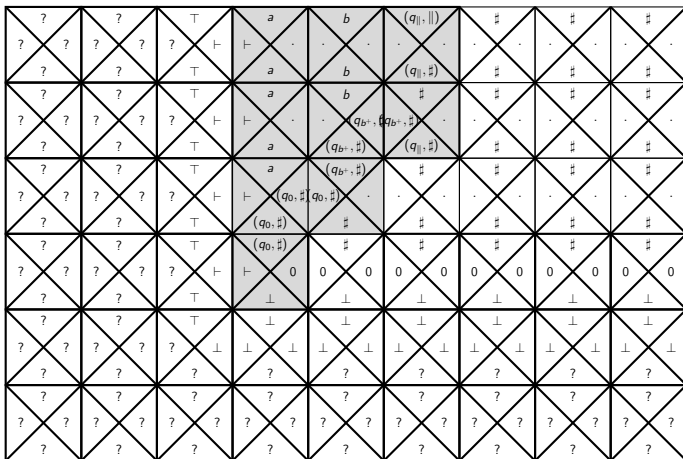
Which tilings ?

We **forbid** tiles with an halting state q_f .

Which tilings ?

We **forbid** tiles with an halting state q_f .

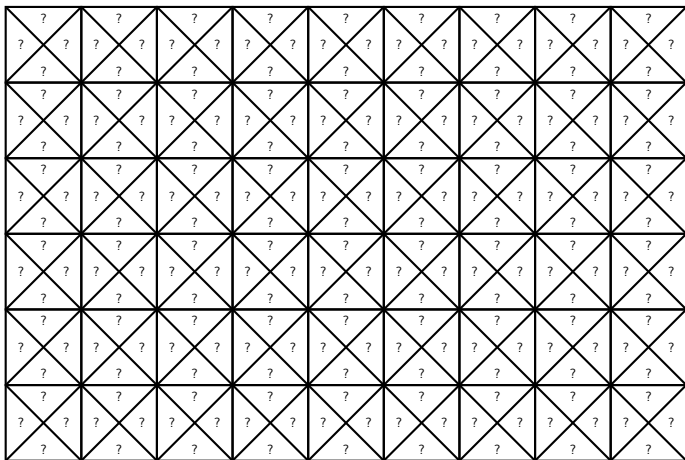
If \mathcal{M} does not halt on the empty input, we have a tiling.



Which tilings ?

We **forbid** tiles with an halting state q_f .

If \mathcal{M} does not halt on the empty input, we have a tiling. But...



The Origin Constrained Domino problem

What we have not proven:

Not-Yet-Theorem

The Domino problem is undecidable on \mathbb{Z}^2 .

The Origin Constrained Domino problem

What we have not proven:

Not-Yet-Theorem

The Domino problem is undecidable on \mathbb{Z}^2 .

What we have proven:

Theorem (Kahr, Moore & Wang 1962, Büchi 1962)

The Origin Constrained Domino problem is undecidable on \mathbb{Z}^2 .

where

Origin Constrained Domino problem

Input: A finite set of Wang tiles τ , a tile $t \in \tau$

Output: **Yes** if there exists a valid tiling by τ with t at the origin, **No** otherwise.

How to initialize computations ?

Build one infinite in time and space computation zone?

- ▶ **Compactness** \Rightarrow we cannot force one given tile to appear exactly once in every valid tiling

How to initialize computations ?

Build one infinite in time and space computation zone?

- ▶ **Compactness** \Rightarrow we cannot force one given tile to appear exactly once in every valid tiling

Build arbitrarily big computation zones?

- ▶ **Compactness** \Rightarrow if we have arbitrarily big *rectangles* in our tilings, then we also have a tiling with no rectangle.

How to initialize computations ?

Build one infinite in time and space computation zone?

- ▶ **Compactness** \Rightarrow we cannot force one given tile to appear exactly once in every valid tiling

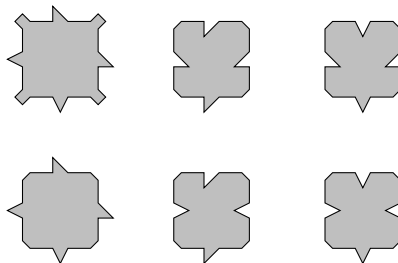
Build arbitrarily big computation zones?

- ▶ **Compactness** \Rightarrow if we have arbitrarily big *rectangles* in our tilings, then we also have a tiling with no rectangle.

One solution: hierarchy of computation zones (thus arbitrarily big zones) that intersect a lot.

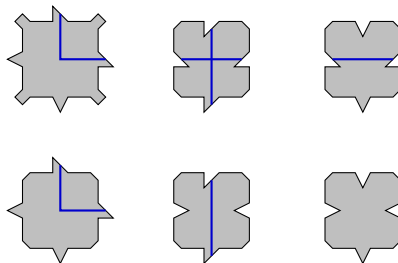
Robinson tileset

The Robinson tileset, where tiles can be rotated and reflected.



Robinson tileset

The Robinson tileset, where tiles can be rotated and reflected.



Existence of a valid tiling

Proposition

Robinson's tileset admits at least one valid tiling.

Existence of a valid tiling

Proposition

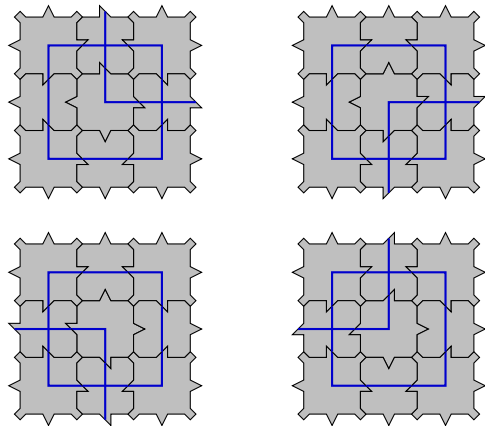
Robinson's tileset admits at least one valid tiling.

Proof:

- We can build arbitrarily large patterns (called macro-tiles) with the same structure.
- We thus conclude by compactness.

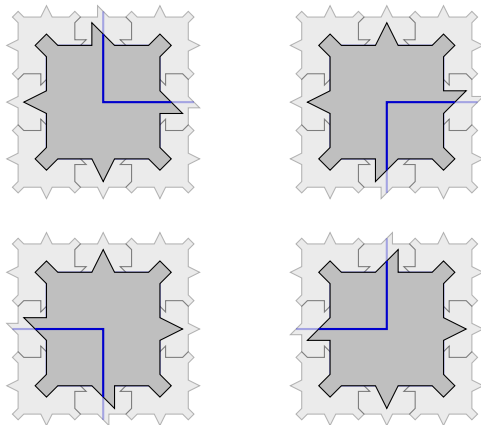
Macro-tiles of level 1


Macro-tiles of level 1.



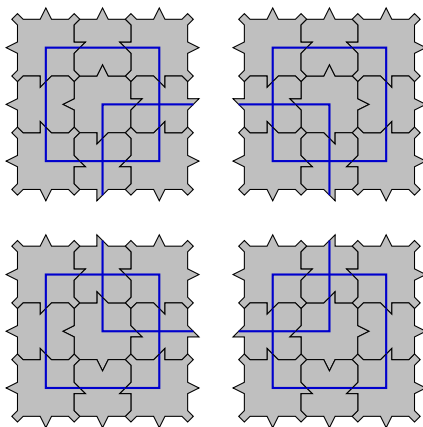
Macro-tiles of level 1

Macro-tiles of level 1.

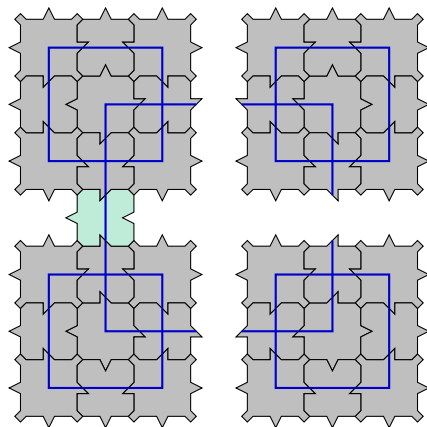


They behave like large .

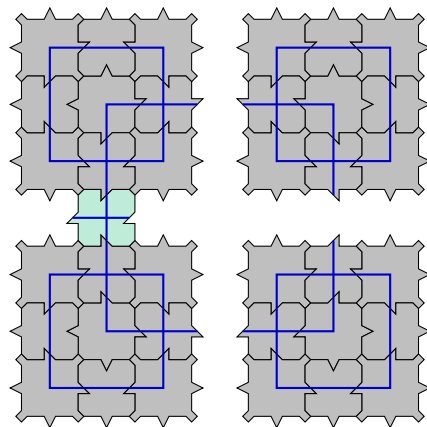
From macro-tiles of level 1 to macro-tiles of level 2



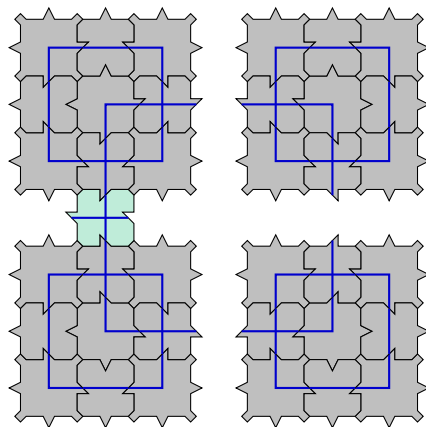
From macro-tiles of level 1 to macro-tiles of level 2



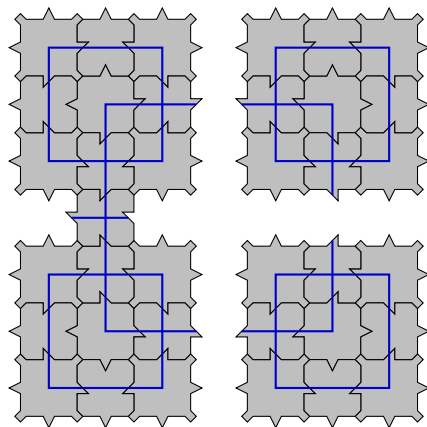
From macro-tiles of level 1 to macro-tiles of level 2



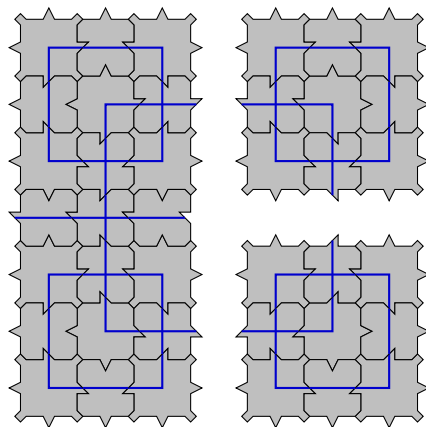
From macro-tiles of level 1 to macro-tiles of level 2



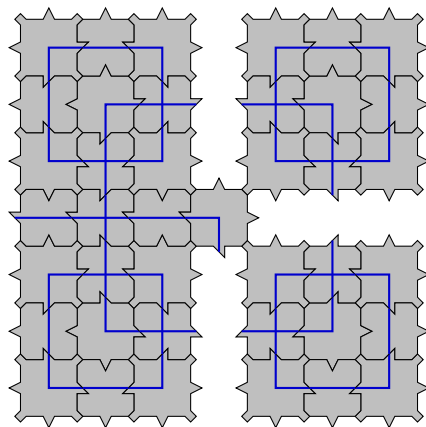
From macro-tiles of level 1 to macro-tiles of level 2



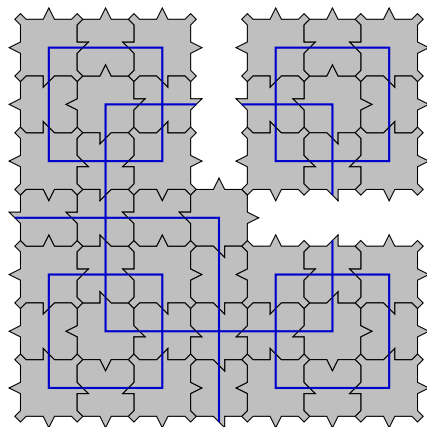
From macro-tiles of level 1 to macro-tiles of level 2



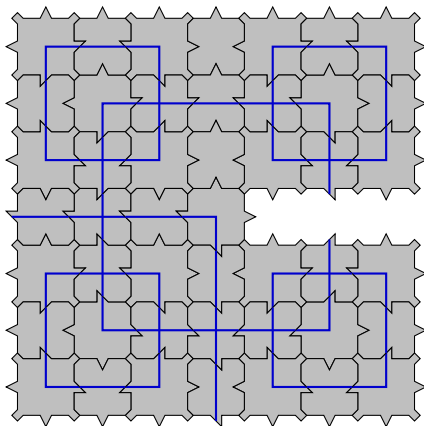
From macro-tiles of level 1 to macro-tiles of level 2



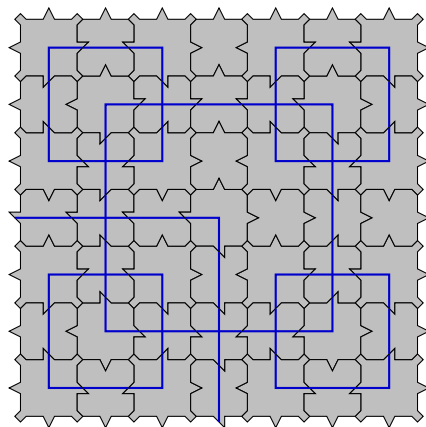
From macro-tiles of level 1 to macro-tiles of level 2



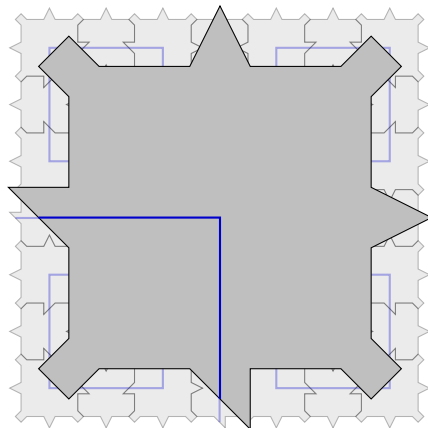
From macro-tiles of level 1 to macro-tiles of level 2



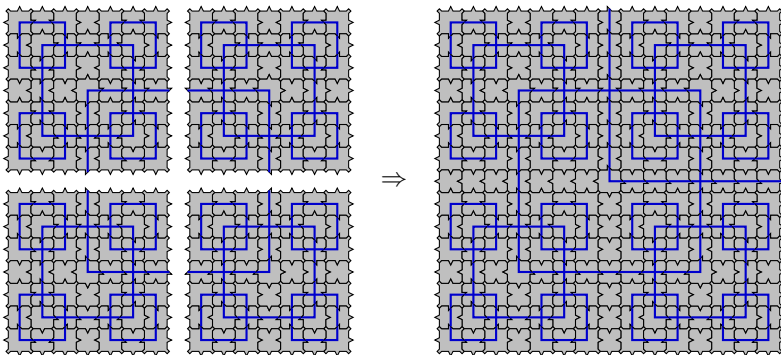
From macro-tiles of level 1 to macro-tiles of level 2



From macro-tiles of level 1 to macro-tiles of level 2

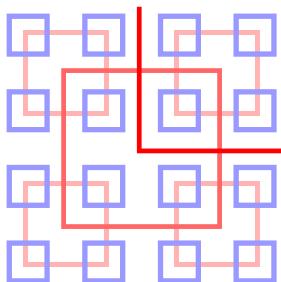


From macro-tiles of level n to macro-tiles of level $n + 1$



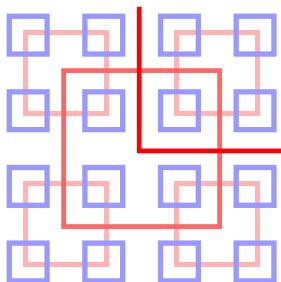
About Robinson's tiling structure

Hierarchy of squares: squares of level n are gathered by 4 to form a square of level $n + 1$



About Robinson's tiling structure

Hierarchy of squares: squares of level n are gathered by 4 to form a square of level $n + 1$

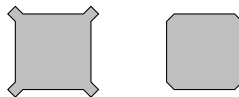


Proposition

The only valid tilings by the Robinson tileset form a hierarchy of squares.

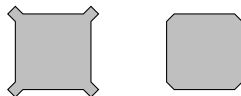
Valid tilings (I)

The two forms in Robinson tileset, cross (bumpy corners) and arms (dented corners).

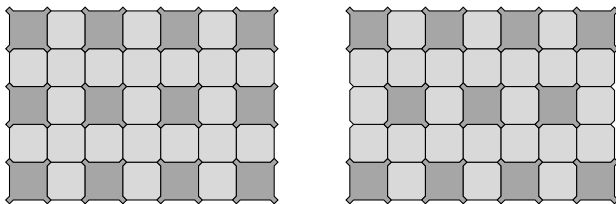


Valid tilings (I)

The two forms in Robinson tileset, cross (bumpy corners) and arms (dented corners).

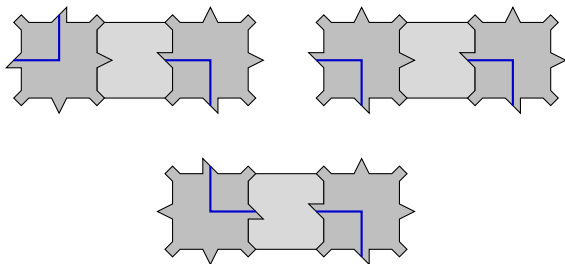


Obviously, two crosses cannot be in contact (neither through an edge nor a vertex) thus a cross must be surrounded by eight arms.



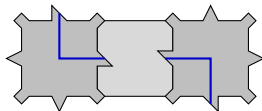
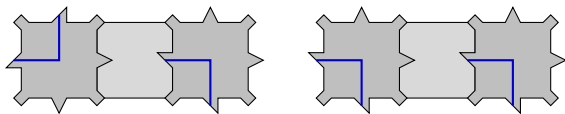
Valid tilings (II)

You cannot have things like

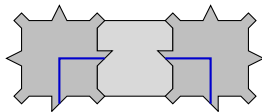


Valid tilings (II)

You cannot have things like

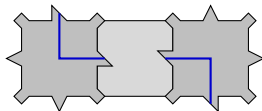
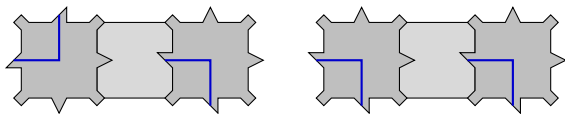


The only possibilities are thus

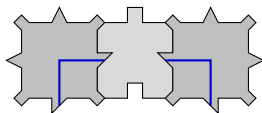


Valid tilings (II)


You cannot have things like

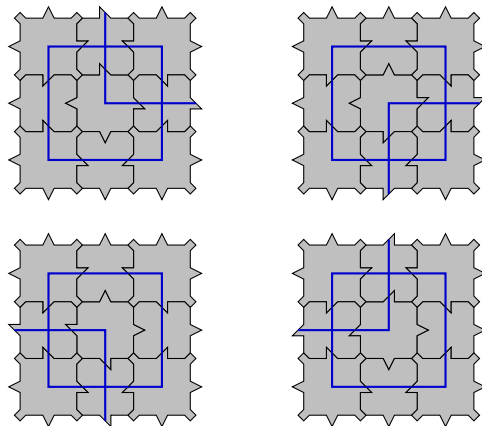



The only possibilities are thus



Valid tilings (III)

So each  is part of a macro tile of level 1

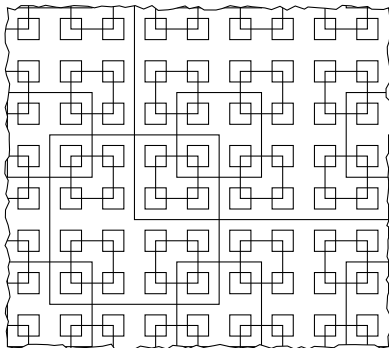


that behaves like a big , and so on...

Undecidability of the Domino Problem (II)

Solution

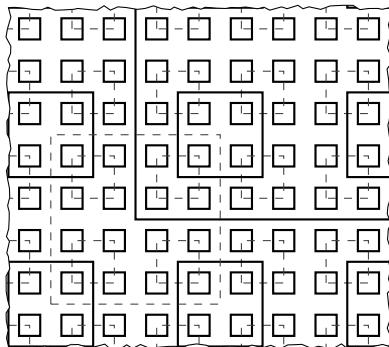
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

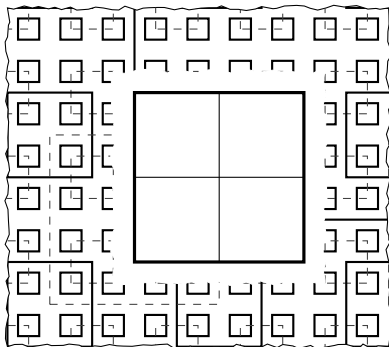
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

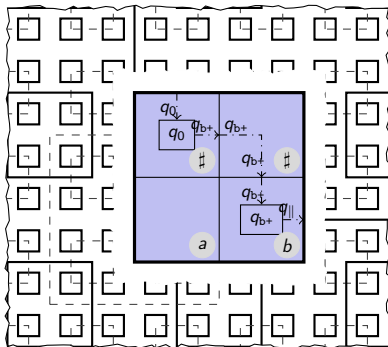
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

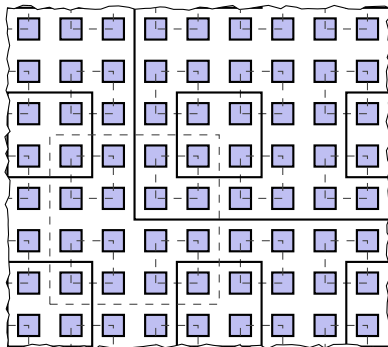
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

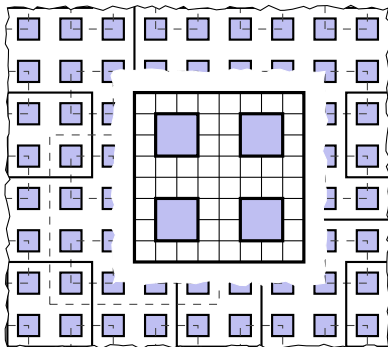
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

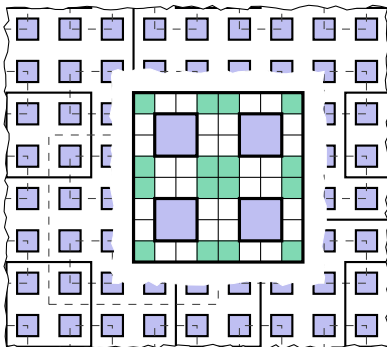
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

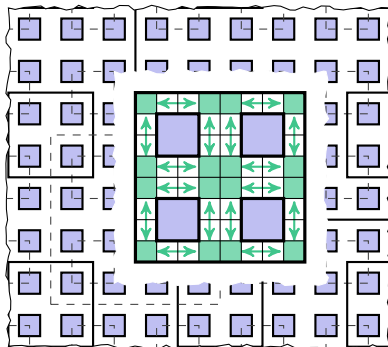
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

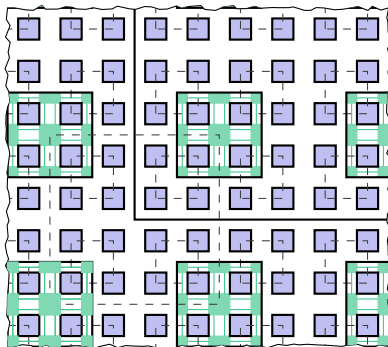
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

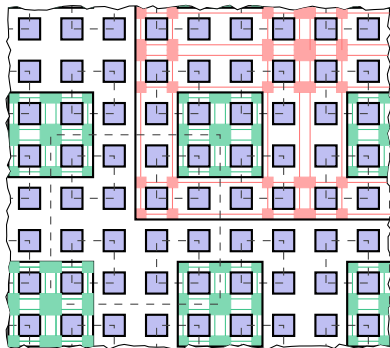
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

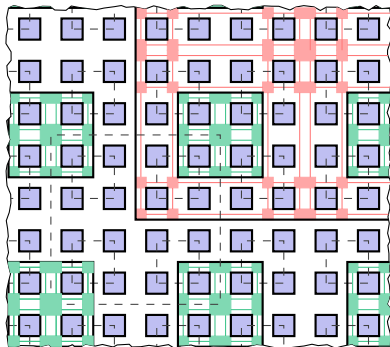
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Theorem (Berger 1966, Robinson 1971)

The Domino Problem is undecidable on \mathbb{Z}^2 .

Outline of the talk.

- 1 Introduction: Symbolic Dynamics and tilings
- 2 Dynamical systems?
- 3 Dimension 2 ($G = \mathbb{Z}^2$)
- 4 Undecidability of DP on \mathbb{Z}^2 , proof I
- 5 Undecidability of DP on \mathbb{Z}^2 , proof II

Sketch of the proof

Idea: encode **piecewise affine maps** inside Wang tiles.

- ▶ Undecidability of the Mortality problem of Turing machines.
- ▶ Undecidability of the Mortality problem of piecewise affine maps.
- ▶ Reduction from the Mortality problem of piecewise affine maps.

Mortality problem of Turing machines

Take \mathcal{M} a deterministic Turing machine with an halting state q_f .

!! configurations of \mathcal{M} do not have finite support !!

A configuration (x, q) is a **non-halting configuration** if it never evolves into the halting state.

Mortality problem of Turing machines

Take \mathcal{M} a deterministic Turing machine with an halting state q_f .

!! configurations of \mathcal{M} do not have finite support !!

A configuration (x, q) is a **non-halting configuration** if it never evolves into the halting state.

Mortality problem of Turing machines

Input: a deterministic Turing machine \mathcal{M} with an halting state.

Output: **Yes** if \mathcal{M} has a non-halting configuration, **No** otherwise.

Theorem (Hooper, 1966)

The Mortality problem of Turing machines is undecidable.

Proof: very technical, uses Minsky 2-counters machines.

Rational piecewise affine maps in \mathbb{R}^2

Take $f_i : U_i \rightarrow \mathbb{R}^2$ for $i \in [1; n]$ some rational affine maps, with U_1, U_2, \dots, U_n disjoint unit squares with integer corners.

Define $f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ with domain $U = \cup_{i=1}^n U_i$ by

$$\vec{x} \mapsto f_i(\vec{x}) \text{ if } \vec{x} \in U_i.$$

A point $\vec{x} \in \mathbb{R}^2$ is an **immortal starting point** for $(f_i)_{i=1\dots n}$ if for every $n \in \mathbb{N}$, the point $f^n(\vec{x})$ lies inside the domain U .

Mortality problem of piecewise affine maps

Input: a system of rational affine maps f_1, f_2, \dots, f_n with disjoint unit squares U_1, U_2, \dots, U_n with integer corners.

Output: **Yes** the system has an immortal starting point, **No** otherwise.

Rational piecewise affine maps and Turing machines (I)

We use the **moving tape** Turing machines model.

Assume that \mathcal{M} has alphabet $A = \{0, 1, \dots, a-1\}$ and states $Q = \{0, 1, \dots, b-1\}$.

Given \mathcal{M} a Turing machine, we construct a system f_1, f_2, \dots, f_n of piecewise affine maps s.t.

- ▶ A configuration of \mathcal{M} is coded by two real numbers.
- ▶ A transition of \mathcal{M} is coded by one f_i .
- ▶ f_1, f_2, \dots, f_n has an immortal starting point if and only if \mathcal{M} has an immortal configuration.

Rational piecewise affine maps and Turing machines (II)

Configuration (x, q) is coded by $(l, r) \in \mathbb{R}^2$ where

$$l = \sum_{i=-1}^{-\infty} M^i x_i$$

and

$$r = Mq + \sum_{i=0}^{\infty} M^{-i} x_i,$$

where M is an integer s.t. $M > a$ and $M > b$.

Rational piecewise affine maps and Turing machines (II)

Configuration (x, q) is coded by $(\ell, r) \in \mathbb{R}^2$ where

$$\ell = \sum_{i=-1}^{-\infty} M^i x_i$$

and

$$r = Mq + \sum_{i=0}^{\infty} M^{-i} x_i,$$

where M is an integer s.t. $M > a$ and $M > b$.

The transition $\delta(q, a) = (q', a', \rightarrow)$ is coded by the affine transformation

$$\begin{pmatrix} \ell \\ r \end{pmatrix} \mapsto \begin{pmatrix} \frac{1}{M} & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \ell \\ r \end{pmatrix} + \begin{pmatrix} a' \\ M(q' - a - Mq) \end{pmatrix}$$

with domain $[0, 1] \times [Mq, Mq + 1]$.

Rational piecewise affine maps and Turing machines (II)

- ▶ A Turing machine \mathcal{M} is transformed into a system f_1, \dots, f_n of rational piecewise affine maps.

Rational piecewise affine maps and Turing machines (II)

- ▶ A Turing machine \mathcal{M} is transformed into a system f_1, \dots, f_n of rational piecewise affine maps.
- ▶ \mathcal{M} has an immortal starting point iff f_1, \dots, f_n has.

Rational piecewise affine maps and Turing machines (II)

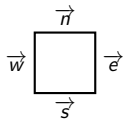
- ▶ A Turing machine \mathcal{M} is transformed into a system f_1, \dots, f_n of rational piecewise affine maps.
- ▶ \mathcal{M} has an immortal starting point iff f_1, \dots, f_n has.

Theorem

The Mortality problem of piecewise affine maps is undecidable.

Rational affine maps inside Wang tiles (I)

Consider $f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ a rational affine map as before. The tile

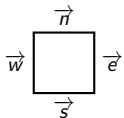


is said to **compute** the function f if

$$f(\vec{n}) + \vec{w} = \vec{s} + \vec{e}.$$

Rational affine maps inside Wang tiles (I)

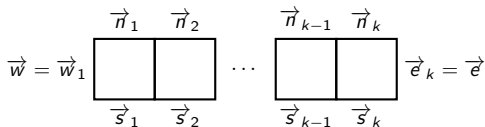
Consider $f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ a rational affine map as before. The tile



is said to **compute** the function f if

$$f(\vec{n}) + \vec{w} = \vec{s} + \vec{e}.$$

And on a row:



$$f\left(\frac{\vec{n}_1 + \cdots + \vec{n}_k}{k}\right) + \frac{1}{k}\vec{w} = \frac{\vec{s}_1 + \cdots + \vec{s}_k}{k} + \frac{1}{k}\vec{e}$$

Rational affine maps inside Wang tiles (II)

For $x \in \mathbb{R}$, a **representation of x** is a sequence of integers $(x_k)_{k \in \mathbb{Z}}$ s.t.

- $\forall k \in \mathbb{Z}, x_k \in \{\lfloor x \rfloor, \lfloor x \rfloor + 1\}$;
- $\forall k \in \mathbb{Z},$

$$\lim_{n \rightarrow \infty} \frac{x_{k-n} + \cdots + x_{k+n}}{2n+1} = x.$$

Rational affine maps inside Wang tiles (II)

For $x \in \mathbb{R}$, a **representation of x** is a sequence of integers $(x_k)_{k \in \mathbb{Z}}$ s.t.

- $\forall k \in \mathbb{Z}, x_k \in \{\lfloor x \rfloor, \lfloor x \rfloor + 1\}$;
- $\forall k \in \mathbb{Z},$

$$\lim_{n \rightarrow \infty} \frac{x_{k-n} + \cdots + x_{k+n}}{2n+1} = x.$$

Define $B_k(x) = \lfloor kx \rfloor - \lfloor (k-1)x \rfloor$ for every $k \in \mathbb{Z}$. Then

$$B(x) = (B_k(x))_{k \in \mathbb{Z}}$$

is the **balanced representation of x** .

Rational affine maps inside Wang tiles (II)

For $x \in \mathbb{R}$, a **representation of x** is a sequence of integers $(x_k)_{k \in \mathbb{Z}}$ s.t.

- $\forall k \in \mathbb{Z}, x_k \in \{\lfloor x \rfloor, \lfloor x \rfloor + 1\}$;
- $\forall k \in \mathbb{Z}$,

$$\lim_{n \rightarrow \infty} \frac{x_{k-n} + \cdots + x_{k+n}}{2n+1} = x.$$

Define $B_k(x) = \lfloor kx \rfloor - \lfloor (k-1)x \rfloor$ for every $k \in \mathbb{Z}$. Then

$$B(x) = (B_k(x))_{k \in \mathbb{Z}}$$

is the **balanced representation of x** .

For $\vec{x} \in \mathbb{R}^2$ and $k \in \mathbb{Z}$, define $B_k(\vec{x})$ coordinate by coordinate.

If \vec{x} is in $U_i = [n, n+1] \times [m, m+1]$, then

$B_k(\vec{x}) \in \{(n, m), (n, m+1), (n+1, m), (n+1, m+1)\}$ for every $k \in \mathbb{Z}$.

Rational affine maps inside Wang tiles (III)

The tile set corresponding to $f_i(\vec{x}) = M\vec{x} + \vec{b}$ consists of tiles

$$\begin{array}{ccc}
 & B_k(\vec{x}) & \\
 f_i(A_{k-1}(\vec{x})) - A_{k-1}(f_i(\vec{x})) & \boxed{} & f_i(A_k(\vec{x})) - A_k(f_i(\vec{x})) \\
 +(k-1)\vec{b} & & +k\vec{b} \\
 & B_k(f_i(\vec{x})) &
 \end{array}$$

for every $k \in \mathbb{Z}$ and $\vec{x} \in U_i$.

Rational affine maps inside Wang tiles (III)

The tile set corresponding to $f_i(\vec{x}) = M\vec{x} + \vec{b}$ consists of tiles

$$\begin{array}{ccc}
 & B_k(\vec{x}) & \\
 f_i(A_{k-1}(\vec{x})) - A_{k-1}(f_i(\vec{x})) & \boxed{} & f_i(A_k(\vec{x})) - A_k(f_i(\vec{x})) \\
 +(k-1)\vec{b} & & +k\vec{b} \\
 & B_k(f_i(\vec{x})) &
 \end{array}$$

for every $k \in \mathbb{Z}$ and $\vec{x} \in U_i$.

Since U_i is bounded and f_i rational, there are **finitely many** tiles !

Rational affine maps inside Wang tiles (IV)

- ▶ A system of rational affine maps f_1, f_2, \dots, f_n defined on U_1, U_2, \dots, U_n with integer corners.
- ▶ Each $f_i \rightsquigarrow$ a finite set of tiles T_i
- ▶ Set of tiles $T = \cup T_i$ with additional markings (every row tiled by a single T_i)
- ▶ T admits a tiling of the plane iff f_1, f_2, \dots, f_n has an immortal point.

Rational affine maps inside Wang tiles (IV)

- ▶ A system of rational affine maps f_1, f_2, \dots, f_n defined on U_1, U_2, \dots, U_n with integer corners.
- ▶ Each $f_i \rightsquigarrow$ a finite set of tiles T_i
- ▶ Set of tiles $T = \cup T_i$ with additional markings (every row tiled by a single T_i)
- ▶ T admits a tiling of the plane iff f_1, f_2, \dots, f_n has an immortal point.

Theorem (Kari, 2007)

The Domino problem is undecidable on \mathbb{Z}^2 .

Domino problem and its variants

Theorem (Kahr, Moore & Wang 1962, Büchi 1962)

The **Origin Constrained Domino problem** is undecidable on \mathbb{Z}^2 .

Theorem (Berger 1966, Robinson 1971, Kari, 2007)

The **Domino problem** is undecidable on \mathbb{Z}^2 .

Theorem (Gurevich & Koryakov, 1972)

The **Periodic Domino problem** is undecidable on \mathbb{Z}^2 .

Theorem (Kari, 1991, Lukkarila 2009)

The **Deterministic Domino problem** is undecidable on \mathbb{Z}^2 .

The Periodic Domino problem

Theorem (Gurevich & Koryakov, 1972)

The Periodic Domino problem is undecidable on \mathbb{Z}^2 .

Proof: on the blackboard, with

$$\tau_{\square} = \left\{ \begin{array}{c} \text{yellow square} \\ \text{red square} \\ \text{yellow/red diagonal} \\ \text{red with vertical dashed line} \\ \text{yellow with horizontal dashed line} \\ \text{yellow/red with both dashed lines} \end{array} \right\}$$

\Rightarrow see G. Theyssier's talk for an example of application.

The Periodic Domino problem

Theorem (Gurevich & Koryakov, 1972)

The Periodic Domino problem is undecidable on \mathbb{Z}^2 .

Proof: on the blackboard, with

$$\tau_{\square} = \left\{ \begin{array}{c} \text{yellow square} \\ \text{red square} \\ \text{yellow triangle (top-left)} \\ \text{red triangle (top-right)} \\ \text{red square with vertical dashed line} \\ \text{yellow square with horizontal dashed line} \\ \text{red square with diagonal dashed line} \end{array} \right\}$$

\Rightarrow see G. Theyssier's talk for an example of application.

Theorem (Kari, 1991, Lukkarila 2009)

The **Deterministic Domino problem** is undecidable on \mathbb{Z}^2 .

\Rightarrow see G. Theyssier's talk for an example of application.

Conclusion

- ▶ Dimension 1: good representation with graphs/matrices for SFTs/sofic subshifts.
- ▶ Dimension 2: much more complicated (encode computational models inside Wang tiles \Rightarrow undecidability results)
- ▶ What about other f.g. groups ?

Conclusion

- ▶ Dimension 1: good representation with graphs/matrices for SFTs/sofic subshifts.
- ▶ Dimension 2: much more complicated (encode computational models inside Wang tiles \Rightarrow undecidability results)
- ▶ What about other f.g. groups ?

Thank you for your attention !!

Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Proof: Consider the G -SFT X_k , where $k = |B_1|$, with alphabet

$$A_3 = \left\{ \triangle, \triangle \begin{array}{c} \bullet \\ \vdots \\ \vdots \end{array}, \triangle \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \end{array} \right\} + \text{rotations}$$

$$A_4 = \left\{ \square, \square \begin{array}{c} \bullet \\ \vdots \\ \vdots \end{array}, \square \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \end{array} \right\} + \text{rotations}$$

$$A_5 = \left\{ \text{pentagon}, \text{pentagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \end{array}, \text{pentagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \end{array}, \text{pentagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array}, \text{pentagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array} \right\} + \text{rotations}$$

$$A_6 = \left\{ \text{hexagon}, \text{hexagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \end{array}, \text{hexagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \end{array}, \text{hexagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array}, \text{hexagon} \begin{array}{c} \bullet \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array} \right\} + \text{rotations and reflections}$$

etc. . .

Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Proof: Take for instance $k = 4$ (for \mathbb{Z}^2 or $BS(m, n)$)

$$A_4 = \left\{ \square, \begin{array}{|c|} \hline \bullet \\ \hline \end{array}, \begin{array}{|c|} \hline \bullet \\ \hline \bullet \\ \hline \end{array} \right\} + \text{rotations}$$

and chose the letter-to-letter map

$$\phi(\square) = 0 \quad \phi\left(\begin{array}{|c|} \hline \bullet \\ \hline \end{array}\right) = \phi\left(\begin{array}{|c|} \hline \bullet \\ \hline \bullet \\ \hline \end{array}\right) = 1$$

Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

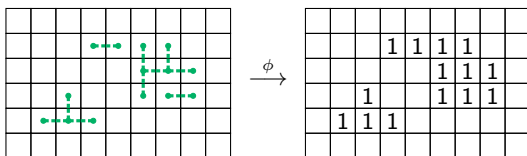
Proof: Take for instance $k = 4$ (for \mathbb{Z}^2 or $BS(m, n)$)

$$A_4 = \left\{ \square, \begin{array}{|c|} \hline \bullet \\ \hline \end{array}, \begin{array}{|c|} \hline \bullet \\ \hline \text{---} \\ \hline \end{array} \right\} + \text{rotations}$$

and chose the letter-to-letter map

$$\phi(\square) = 0 \quad \phi\left(\begin{array}{|c|} \hline \bullet \\ \hline \end{array}\right) = \phi\left(\begin{array}{|c|} \hline \bullet \\ \hline \text{---} \\ \hline \end{array}\right) = 1$$

Green components have even size (handshaking lemma) $\Rightarrow \phi(X_k) \subseteq X_{\text{even}}$



Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.

			1	1	1	1			
					1	1	1		
		1			1	1	1		
	1	1	1						

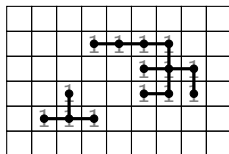
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .

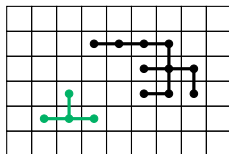
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.

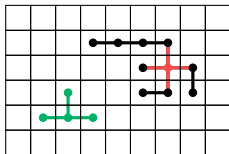
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.

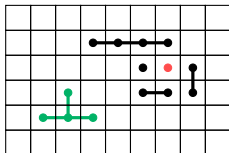
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.

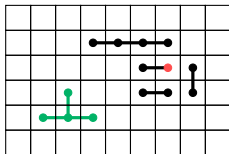
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.
- ▶ In $\mathcal{T} \setminus \{v\}$, odd number of trees with odd cardinality: connect v to them.

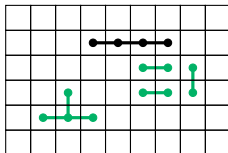
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.
- ▶ In $\mathcal{T} \setminus \{v\}$, odd number of trees with odd cardinality: connect v to them.
- ▶ Iterate the process to get rid of all vertices with even degree, and conclude by compactness.

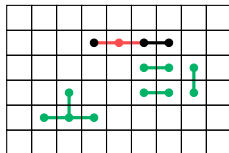
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.
- ▶ In $\mathcal{T} \setminus \{v\}$, odd number of trees with odd cardinality: connect v to them.
- ▶ Iterate the process to get rid of all vertices with even degree, and conclude by compactness.

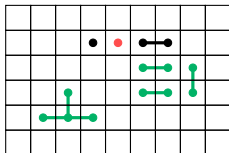
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.
- ▶ In $\mathcal{T} \setminus \{v\}$, odd number of trees with odd cardinality: connect v to them.
- ▶ Iterate the process to get rid of all vertices with even degree, and conclude by compactness.

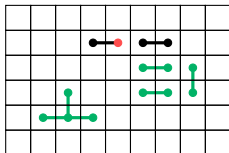
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.
- ▶ In $\mathcal{T} \setminus \{v\}$, odd number of trees with odd cardinality: connect v to them.
- ▶ Iterate the process to get rid of all vertices with even degree, and conclude by compactness.

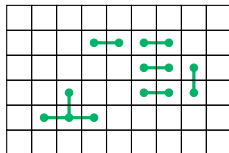
Example: the even shift

$$X_{\text{even}} = \{x \in \{0, 1\}^G \mid \text{finite CC of 1's have even size}\}.$$

Proposition

The even shift X_{even} is sofic for every f.g. group G .

Conversely, for some $x \in X_{\text{even}}$, consider \mathcal{C} a maximal CC of 1.



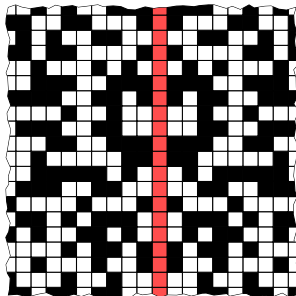
- ▶ Chose \mathcal{T} a tree covering of \mathcal{C} .
- ▶ If all vertices in \mathcal{T} have odd degree, then we are done.
- ▶ Otherwise, delete a vertex v with even degree \Rightarrow forest of CC of 1.
- ▶ In $\mathcal{T} \setminus \{v\}$, odd number of trees with odd cardinality: connect v to them.
- ▶ Iterate the process to get rid of all vertices with even degree, and conclude by compactness.

Mirror subshift in \mathbb{Z}^2

Let $A = \{ \square, \blacksquare, \color{red}\square \}$ and $X_{\text{mirror}} = X_{F_{\text{mirror}}} \subset A^{\mathbb{Z}^2}$ where

$$F_{\text{mirror}} = \left\{ \begin{array}{|c|} \hline \square \\ \hline \color{red}\square \\ \hline \end{array}, \begin{array}{|c|} \hline \blacksquare \\ \hline \color{red}\square \\ \hline \end{array}, \begin{array}{|c|} \hline \color{red}\square \\ \hline \square \\ \hline \end{array}, \begin{array}{|c|} \hline \color{red}\square \\ \hline \blacksquare \\ \hline \end{array} \right\} \cup \bigcup_{w \in A^*} \{ \color{red}\square w \color{red}\square, \blacksquare w \color{red}\square \tilde{w} \square, \square w \color{red}\square \tilde{w} \blacksquare \}$$

where \tilde{w} denotes the mirror image of the word w .



The mirror subshift is not sofic

