Lecture 1: The Domino problem on groups, part I.

CANT 2016, CIRM (Marseille)

Nathalie Aubrun

LIP, ENS de Lyon, CNRS

29th November 2016

Introduction

Objectives of this talk. . .

- Define the Domino problem (**DP**).
- Show the two main techniques to prove undecidability of **DP** on $\mathbb{Z}^2$

Outline of the talk.

Configurations and Subshifts (I)

- Let A be a finite alphabet, G be a finitely generated group.
- Colorings $x : G \to A$ are called **configurations**.
- Endowed with the prodiscrete topology A^G is a **compact** and **metrizable** set.
- **Cylinders** form a clopen basis

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

- A **pattern** is a finite intersection of cylinders, or equivalently a finite configuration $p : S \to A$
- 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 $\Gamma(G, S)$.

Configurations and Subshifts (II)

The **shift** action $\sigma : G \times A^G \to A^G$ is given by

$$(\sigma_g(x))_h = x_{g^{-1}h}.$$

The dynamical system (A^G, σ) is called the ***G*-fullshift over *A***.

Definition

A ***G*-subshift** is a closed and σ-invariant subset $X \subset A^G$.

Configurations and Subshifts (II)

The **shift** action $\sigma : G \times A^G \to A^G$ is given by

$$(\sigma_g(x))_h = x_{g^{-1}h}.$$

The dynamical system (A^G, σ) is called the **G-fullshift over A**.

Definition

A **G-subshift** is a closed and σ-invariant subset $X \subset A^G$.

A pattern $p \in A^S$ **appears** in a configuration $x \in A^G$ if $(\sigma_g(x))_S = p$ for some $g \in G$.

Proposition

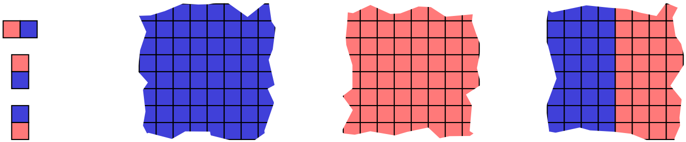
X is a G-subshift iff there exists a set $\mathcal{F}$ of forbidden patterns s.t.

$$X = X_{\mathcal{F}} := \left\{x \in A^G \mid \text{ no pattern of } \mathcal{F} \text{ appears in } x\right\}.$$

Subshifts of finite type

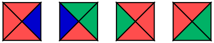
A G-subshift X is **of finite type** (G-SFT) if there exists a finite set of forbidden patterns $\mathcal{F}$ that defines it: $X = X_{\mathcal{F}}$.

Example:



SFTs and Wang tiles

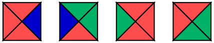
Fix G a f.g. group and S a generating set for G. Wang tiles $\approx$ polygons with colored $2|S|$ edges.



Neighbourhood rule

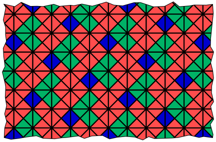
SFTs and Wang tiles

Fix G a f.g. group and S a generating set for G. Wang tiles $\approx$ polygons with colored $2|S|$ edges.



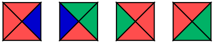
Neighbourhood rule

X_τ set of valid tilings by τ



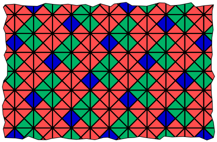
SFTs and Wang tiles

Fix G a f.g. group and S a generating set for G. Wang tiles $\approx$ polygons with colored $2|S|$ edges.

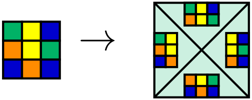


Neighbourhood rule

X_τ set of valid tilings by τ



SFT $\approx X_\tau$



The Domino problem on groups

Fix G a f.g. group and S a generating set for G.

Domino problem on G

Input: A finite set of Wang tiles τ on S
Output: **Yes** if there exists a valid tiling by τ, **No** otherwise.

The Domino problem on groups

Fix G a f.g. group and S a generating set for G.

Domino problem on G

Input: A finite set of Wang tiles τ on S
Output: **Yes** if there exists a valid tiling by τ, **No** otherwise.

Question

Which f.g. groups have decidable Domino Problem ?

The Domino problem on groups

Fix G a f.g. group and S a generating set for G.

Domino problem on G

Input: A finite set of Wang tiles τ on S
Output: **Yes** if there exists a valid tiling by τ, **No** otherwise.

Question

Which f.g. groups have decidable Domino Problem ?

$\rightarrow$ group property, quasi-isometry invariant.

Outline of the talk.

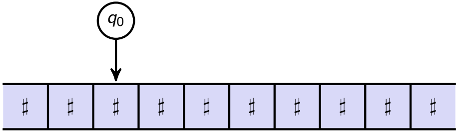
Sketch of the proof

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

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

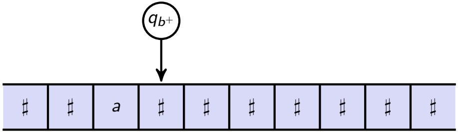
Turing machines

$\delta(q,x)$		Symbol x			
		a	b	$\|$	$\sharp$
State q	q_0	$\bot$	$\bot$	$\bot$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\bot$	$(q_{b^{++}}, a, \rightarrow)$	$\bot$	$\bot$
	q_{b^+}	$\bot$	$\bot$	$\bot$	$(q_{\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\bot$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\bot$
	$q_{\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|}, b, \leftarrow)$	$(q_{\|}, \|, \leftarrow)$	$(q_{\|}, \|, \cdot)$



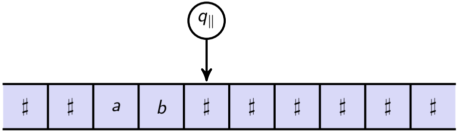
Turing machines

$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\bot$	$\bot$	$\bot$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\bot$	$(q_{b^{++}}, a, \rightarrow)$	$\bot$	$\bot$
	q_{b^+}	$\bot$	$\bot$	$\bot$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\bot$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\bot$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



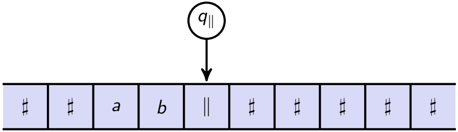
Turing machines

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



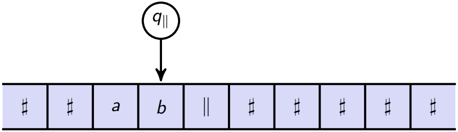
Turing machines

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



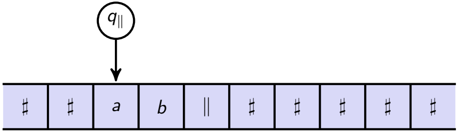
Turing machines

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



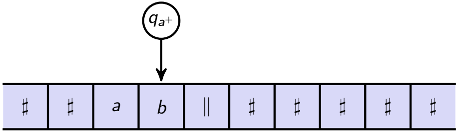
Turing machines

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



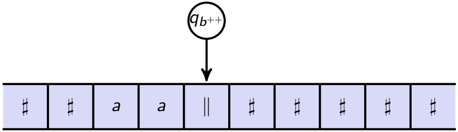
Turing machines

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



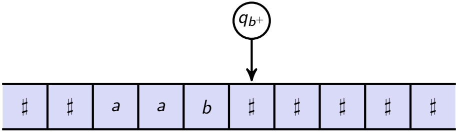
Turing machines

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



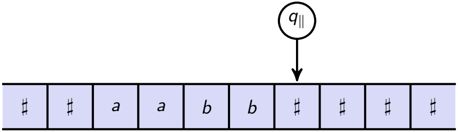
Turing machines

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



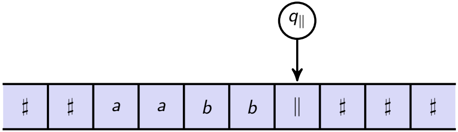
Turing machines

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



Turing machines

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



Turing machines

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

Theorem (Turing, 1936)

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

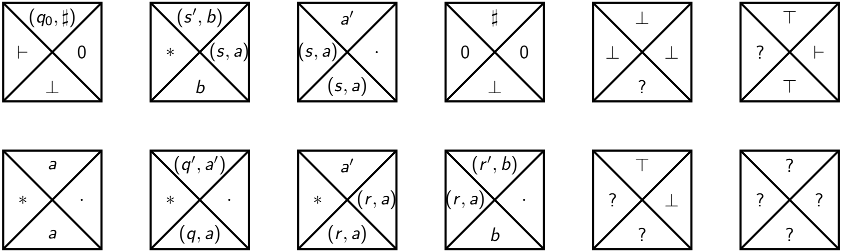
Theorem

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:

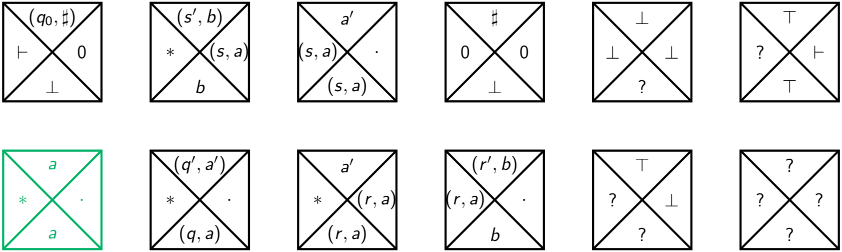
- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

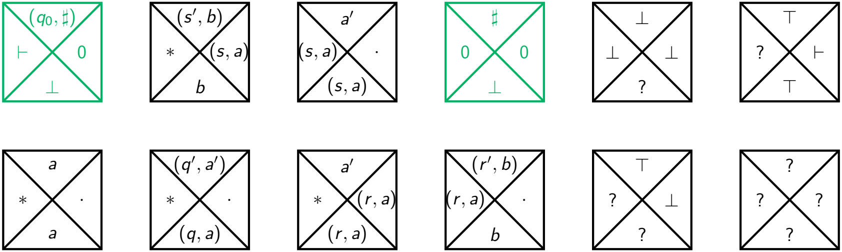
- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

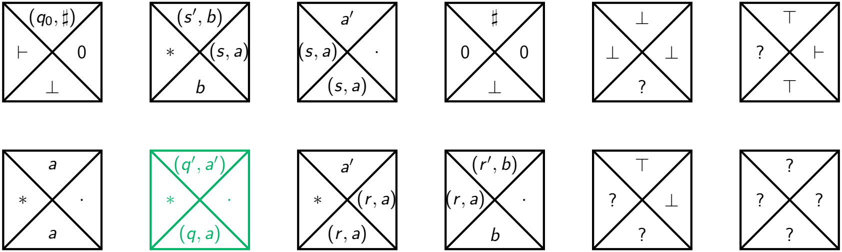
- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

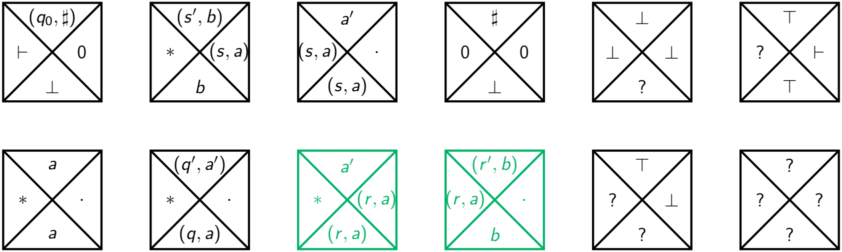
- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

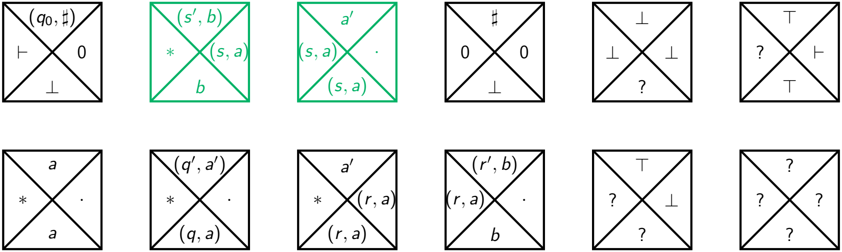
- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

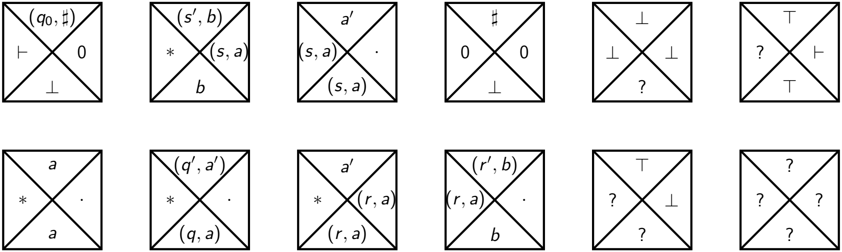
- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



Turing machines and Wang tiles

Encode Turing machine computations inside Wang tiles:

- no computation head
- initial configuration $({}^{\infty}\sharp^{\infty}, q_0)$
- $\delta(q, a) = (q', a', .)$
- $\delta(r, a) = (r', a', \rightarrow)$
- $\delta(s, a) = (s', a', \leftarrow)$



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

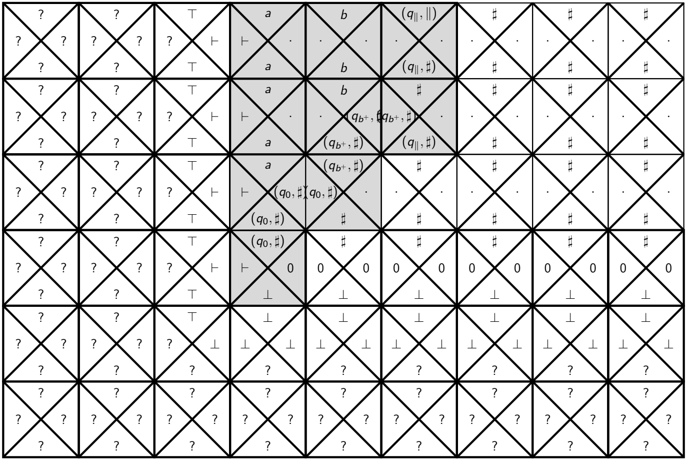
Which tilings ?

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

Which tilings ?

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

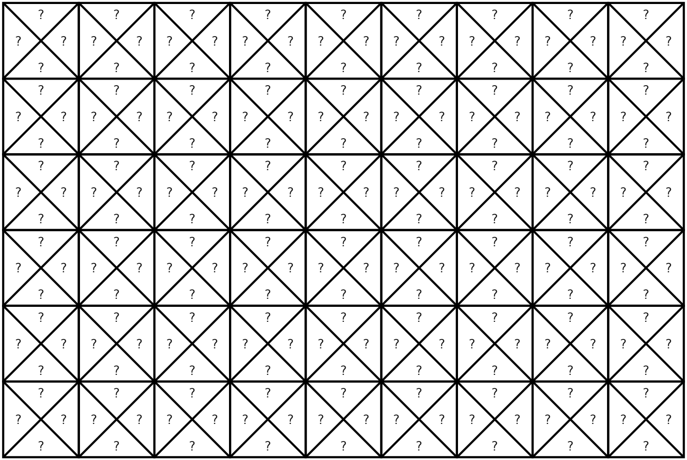
If $\mathcal{M}$ does not halt on the empty input, we have a tiling.



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

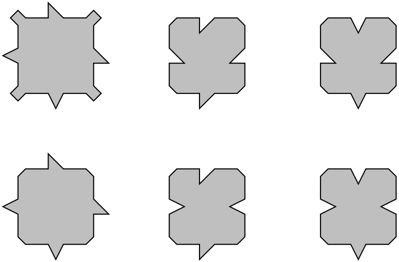
Build arbitrarily big computation zones?

- **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 arbritrarily big zones) that intersect a lot.

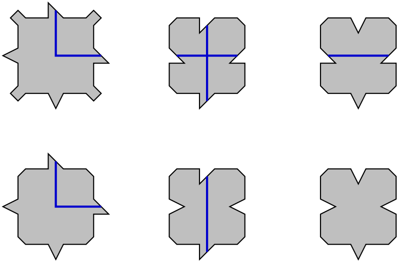
Robinson tileset

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

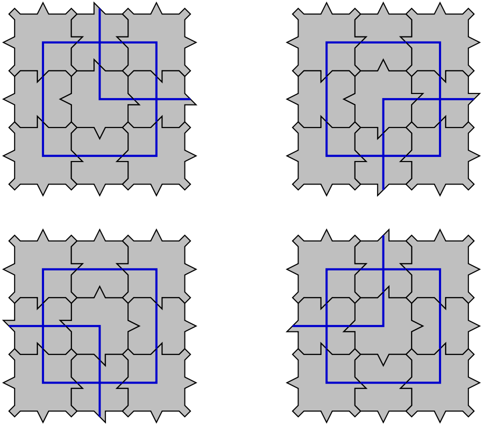
Robinson's tileset admits at least one valid tiling.

Proof:

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

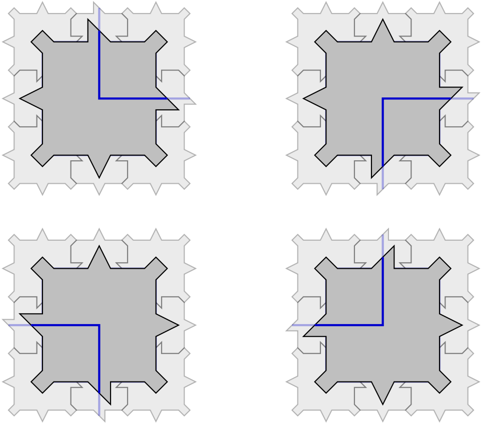
Macro-tiles of level 1

Macro-tiles of level 1.



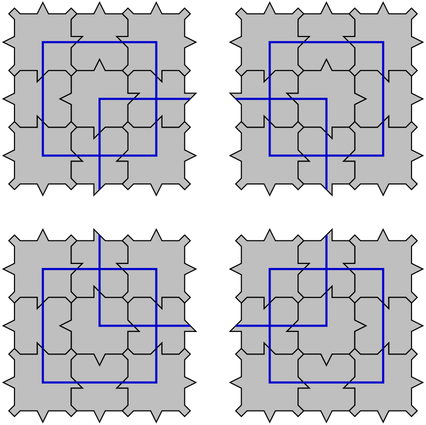
Macro-tiles of level 1

Macro-tiles of level 1.

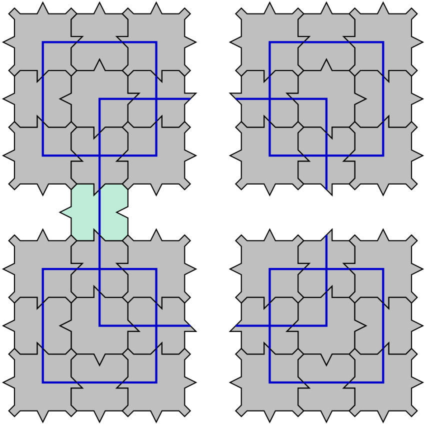


They behave like large .

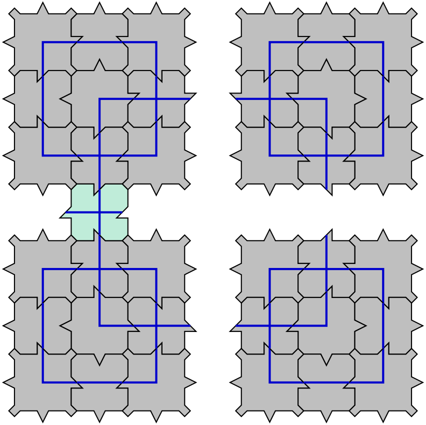
From macro-tiles of level 1 to macro-tiles of level 2



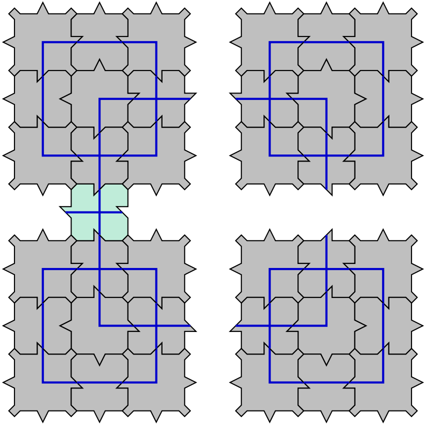
From macro-tiles of level 1 to macro-tiles of level 2



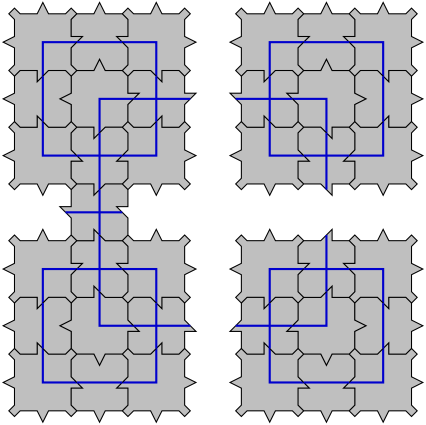
From macro-tiles of level 1 to macro-tiles of level 2



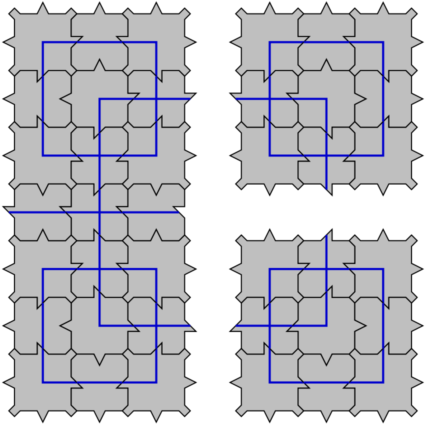
From macro-tiles of level 1 to macro-tiles of level 2



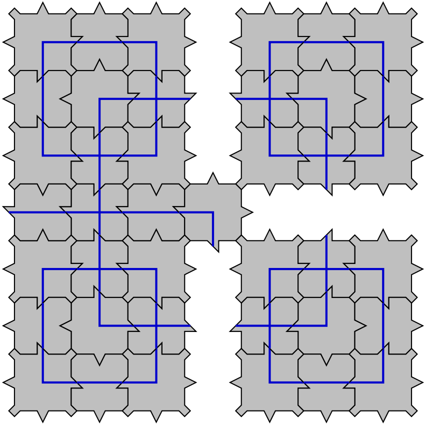
From macro-tiles of level 1 to macro-tiles of level 2



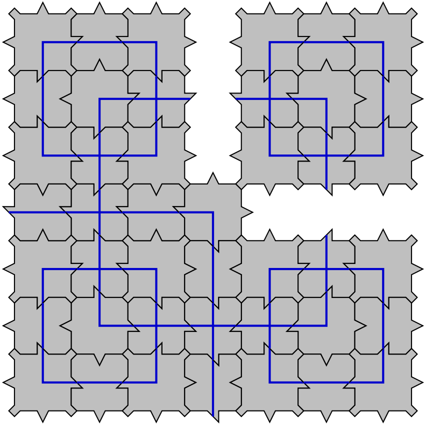
From macro-tiles of level 1 to macro-tiles of level 2



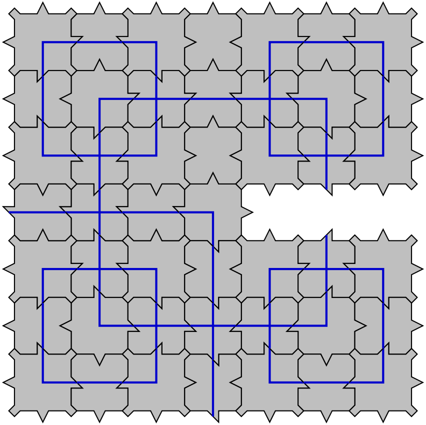
From macro-tiles of level 1 to macro-tiles of level 2



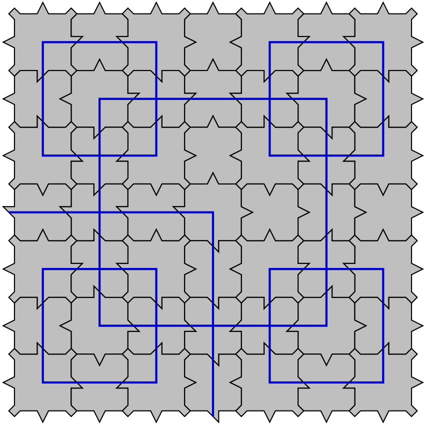
From macro-tiles of level 1 to macro-tiles of level 2



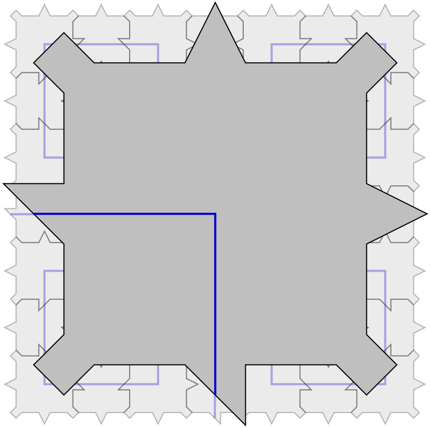
From macro-tiles of level 1 to macro-tiles of level 2



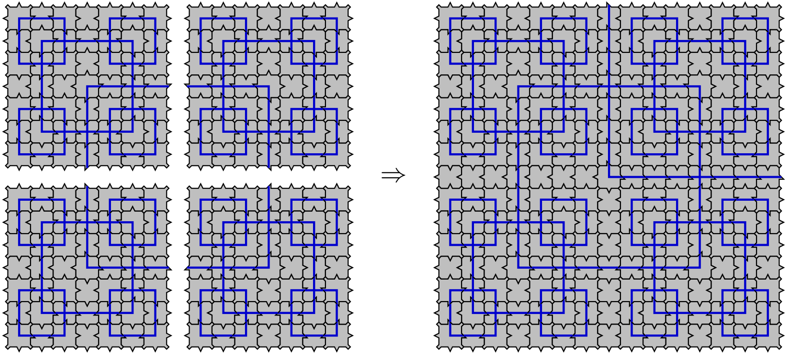
From macro-tiles of level 1 to macro-tiles of level 2



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

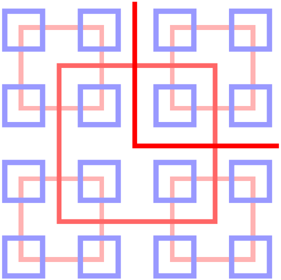


From macro-tiles of level n to macro-tiles of level $n+1$



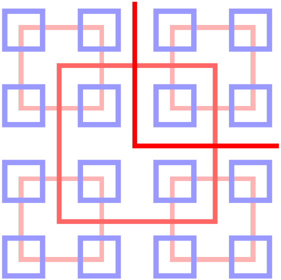
About Robinson's tiling structure

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 $n+1$

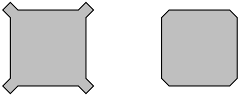


Proposition

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

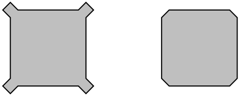
Valid tilings (I)

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

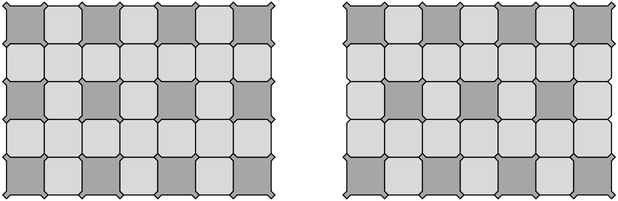


Valid tilings (I)

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

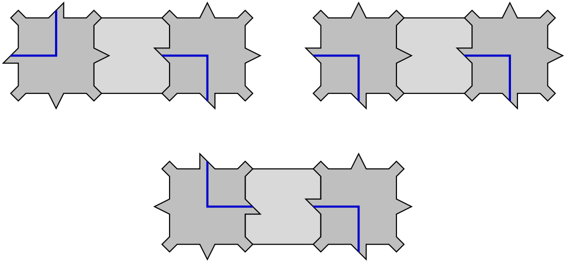


Obviously, two crosses cannot be in contact (neither through an edge nor a vertex) thus a cross must be surrounded by eight arms.



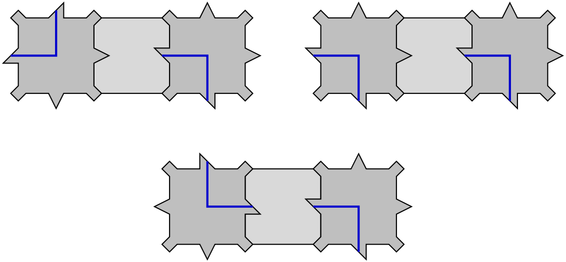
Valid tilings (II)

You cannot have things like

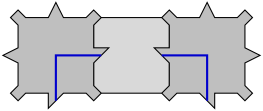


Valid tilings (II)

You cannot have things like

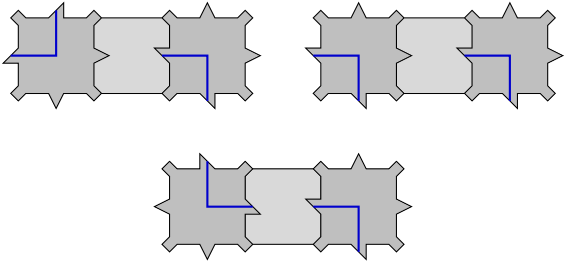


The only possibilities are thus

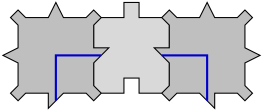


Valid tilings (II)

You cannot have things like

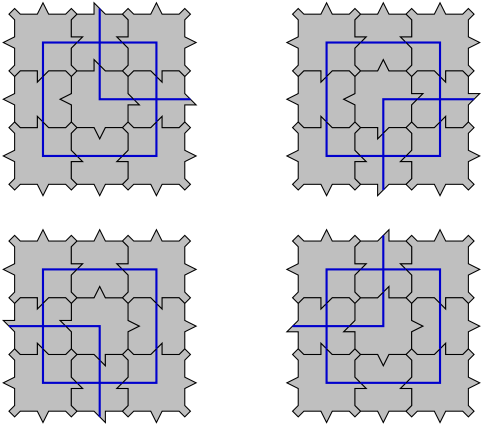


The only possibilities are thus



Valid tilings (III)

So each  is part of a macro tile of level 1

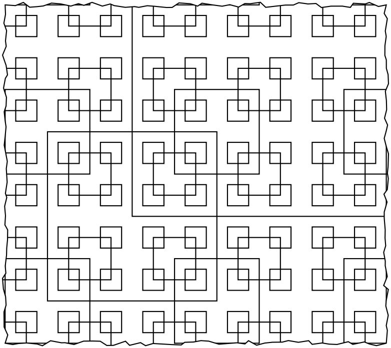


that behaves like a big , and so on. . .

Undecidability of the Domino Problem (II)

Solution

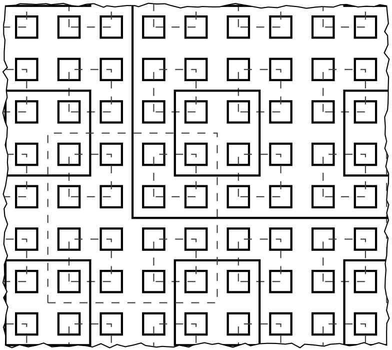
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

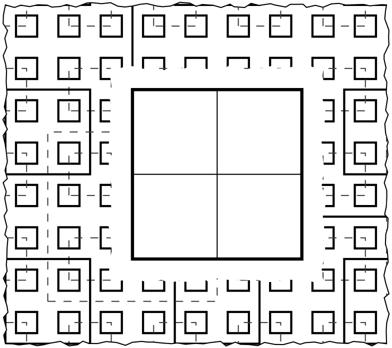
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

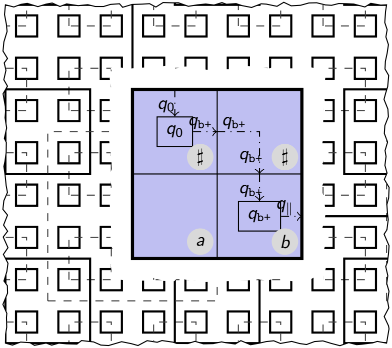
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

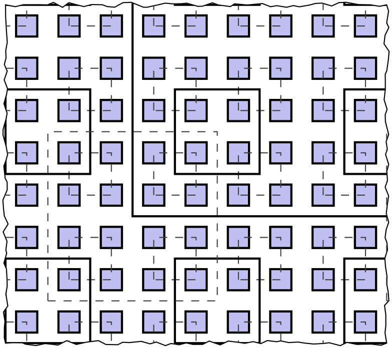
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

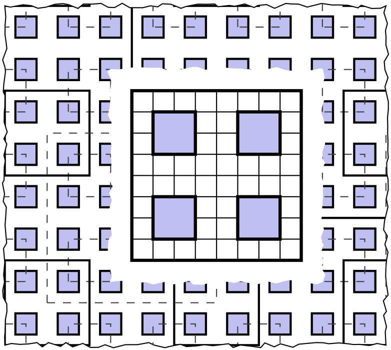
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

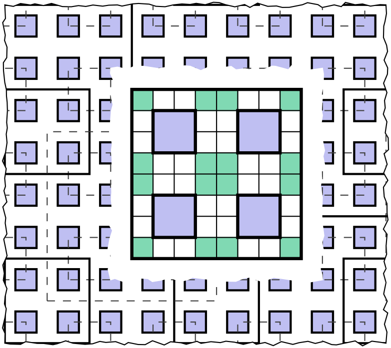
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

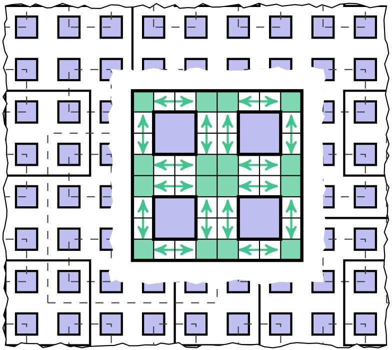
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

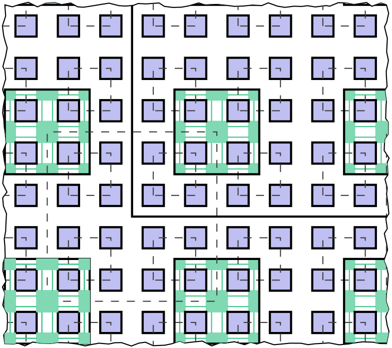
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

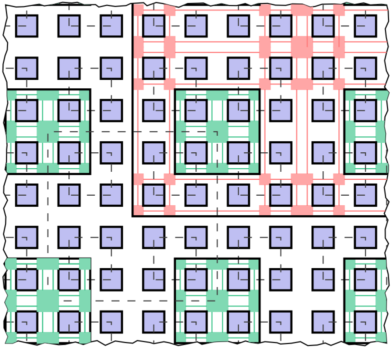
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



Undecidability of the Domino Problem (II)

Solution

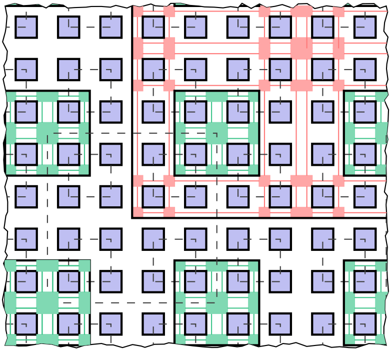
Embed Turing machine computations inside the hierarchy of squares given by Robinson's tiling.



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.

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

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

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

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

Mortality problem of piecewise affine maps

Input: a system of rational affine maps $f_1, f_2, \ldots, f_n$ with disjoint unit squares $U_1, U_2, \ldots, U_n$ 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 $A = \{0, 1, \ldots, a-1\}$ and states $Q = \{0, 1, \ldots, b-1\}$.

Given $\mathcal{M}$ a Turing machine, we construct a system $f_1, f_2, \ldots, f_n$ 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.
- $f_1, f_2, \ldots, f_n$ 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 $(\ell, r) \in \mathbb{R}^2$ 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 $M > b$.

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 $f_1, \ldots, f_n$ of rational piecewise affine maps.

Rational piecewise affine maps and Turing machines (II)

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

Rational piecewise affine maps and Turing machines (II)

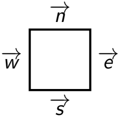
- A Turing machine $\mathcal{M}$ is transformed into a system $f_1, \ldots, f_n$ of rational piecewise affine maps.
- $\mathcal{M}$ has an immortal starting point iff $f_1, \ldots, f_n$ has.

Theorem

The Mortality problem of piecewise affine maps is undecidable.

Rational affine maps inside Wang tiles (I)

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

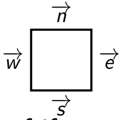


is said to **compute** the function f if

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

Rational affine maps inside Wang tiles (I)

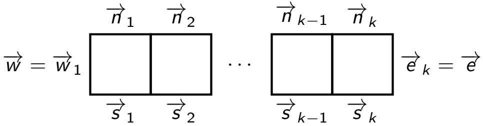
Consider $f : \mathbb{R}^2 \to \mathbb{R}^2$ a rational affine map as before. The tile



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.

- $\forall k \in \mathbb{Z}, x_k \in \{\lfloor x \rfloor, \lfloor x \rfloor + 1\}$;
- $\forall k \in \mathbb{Z}$,
$$\lim_{n\to\infty} \frac{x_{k-n} + \cdots + x_{k+n}}{2n+1} = x.$$

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.

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

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

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.

- $\forall k \in \mathbb{Z}, x_k \in \{\lfloor x \rfloor, \lfloor x \rfloor + 1\}$;
- $\forall k \in \mathbb{Z}$,
$$\lim_{n\to\infty} \frac{x_{k-n} + \cdots + x_{k+n}}{2n+1} = x.$$

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

If $\overrightarrow{x}$ is in $U_i = [n, n+1] \times [m, m+1]$, then
$B_k(\overrightarrow{x}) \in \{(n,m), (n, m+1), (n+1, m), (n+1, m+1)\}$ for every $k \in \mathbb{Z}$.

Rational affine maps inside Wang tiles (III)

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

$$\begin{array}{ccc} & B_k(\overrightarrow{x}) & \\ \begin{array}{r} f_i(A_{k-1}(\overrightarrow{x})) - A_{k-1}(f_i(\overrightarrow{x})) \\ +(k-1)\overrightarrow{b} \end{array} & \square & \begin{array}{l} f_i(A_k(\overrightarrow{x})) - A_k(f_i(\overrightarrow{x})) \\ +k\overrightarrow{b} \end{array} \\ & B_k(f_i(\overrightarrow{x})) & \end{array}$$

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

Rational affine maps inside Wang tiles (III)

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

Top: $B_k(\overrightarrow{x})$

Left: $f_i(A_{k-1}(\overrightarrow{x})) - A_{k-1}(f_i(\overrightarrow{x})) + (k-1)\overrightarrow{b}$

Right: $f_i(A_k(\overrightarrow{x})) - A_k(f_i(\overrightarrow{x})) + k\overrightarrow{b}$

Bottom: $B_k(f_i(\overrightarrow{x}))$

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

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 $f_1, f_2, \ldots, f_n$ defined on $U_1, U_2, \ldots, U_n$ 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 $f_1, f_2, \ldots, f_n$ has an immortal point.

Rational affine maps inside Wang tiles (IV)

- A system of rational affine maps $f_1, f_2, \ldots, f_n$ defined on $U_1, U_2, \ldots, U_n$ 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 $f_1, f_2, \ldots, f_n$ has an immortal point.

Theorem (Kari, 2007)

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

Conclusion

- Two proofs of the undecidability of **DP** on $\mathbb{Z}^2$.
- Encode a small computational model inside Wang tiles.
- What about f.g. groups ?

Conclusion

- Two proofs of the undecidability of **DP** on $\mathbb{Z}^2$.
- Encode a small computational model inside Wang tiles.
- What about f.g. groups ?

Thank you for your attention !!