



HAL
open science

Slopes of multidimensional subshifts

Emmanuel Jeandel, Etienne Moutot, Pascal Vanier

► **To cite this version:**

Emmanuel Jeandel, Etienne Moutot, Pascal Vanier. Slopes of multidimensional subshifts. *Theory of Computing Systems*, 2020, 64 (1), pp.35-61. 10.1007/s00224-019-09931-1 . hal-02158012

HAL Id: hal-02158012

<https://inria.hal.science/hal-02158012v1>

Submitted on 17 Jun 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Slopes of multidimensional subshifts

Emmanuel Jeandel · Etienne Moutot ·
Pascal Vanier

Received: date / Accepted: date

Abstract In this paper we study the directions of periodicity of multidimensional subshifts of finite type (SFTs) and of multidimensional effectively closed and sofic subshifts. A configuration of a subshift has a slope of periodicity if it is periodic in exactly one direction, the slope representing that direction. In this paper, we prove that Σ_1^0 sets of non-commensurable \mathbb{Z}^2 vectors are exactly the sets of slopes of 2D SFTs and that Σ_2^0 sets of non-commensurable vectors are exactly the sets of slopes of 3D SFTs, and exactly the sets of slopes of 2D and 3D sofic and effectively closed subshifts.

Keywords subshifts · SFTs · computability · slopes · periodicity

Introduction

A d -dimensional subshift is a set of colorings of \mathbb{Z}^d by a finite number of colors containing no pattern from a family of forbidden patterns. Subshifts may be seen as discretizations of continuous dynamical systems: if X is a compact space and there are d commuting continuous actions ϕ_1, \dots, ϕ_d on X , one can

This work was supported by grants TARMAC ANR 12 BS02 007 01 and CoCoGro ANR 16 CE40 0005.

Emmanuel Jeandel
Université de Lorraine, CNRS, Inria, LORIA, Nancy, France
E-mail: emmanuel.jeandel@loria.fr

Etienne Moutot
LIP, ENS de Lyon – CNRS – INRIA – UCBL – Université de Lyon,
6 allée d’Italie, 69364 Lyon Cedex, France
E-mail: etienne.moutot@ens-lyon.org

Pascal Vanier
Laboratoire d’Algorithmique, Complexité et Logique
Université de Paris-Est, LACL, UPEC, France
E-mail: pascal.vanier@lacl.fr

partition X in a finite number of parts indexed by an alphabet Σ . The orbit of a point $x \in X$ maps to a coloring y of \mathbb{Z}^d where $y(v)$ corresponds to the partition where $\phi^v(x)$ lies.

If the family of forbidden patterns is finite, the subshift is said to be of finite type (SFT for short). In dimension 1, most problems on SFTs are easy in a computational sense, since SFTs correspond to bi-infinite walks on finite automata. For instance, in dimension 1, detecting whether an SFT is non-empty is decidable since it suffices to detect if there exists a cycle in the corresponding automaton [19], which corresponds to the existence of a periodic configuration.

In higher dimensions however, the situation becomes more involved, and knowing whether an SFT is non-empty becomes undecidable [3, 4]. The proof uses two key results on SFTs: the existence of an aperiodic SFT and an encoding of space-time diagrams of Turing machines. The fact that there exists aperiodic SFTs is not straightforward, and the converse was first conjectured by Wang [26]. Had this conjecture been true, it would have meant the decidability of the emptiness problem for SFTs. Berger [3, 4] proved however that there does exist SFTs containing only aperiodic configurations. Subsequently, many other aperiodic SFTs were constructed [5, 13, 18, 23, 24]. Note that the existence in itself of aperiodic SFTs does not suffice a priori to prove that the emptiness problem is undecidable, one needs in addition to encode some computation in them, usually in the form of Turing machines.

Periodicity has thus been central in the study of SFTs from the beginning, and it has been proved early that knowing whether an SFT is aperiodic is undecidable [9]. In fact, sets of periods constitute a classical conjugacy/isomorphism invariant for subshifts in any dimension. As such, they have been studied extensively, leading for example to an algebraic characterization in dimension 1, see [19] for more details. It also seems that computability theory is the right tool to study dynamical aspects of higher dimensional symbolic dynamical systems [1, 7, 12, 20].

In dimensions $d \geq 2$, one may investigate periodicity from different point of views. Denote $\Gamma_x = \{\mathbf{v} \in \mathbb{Z}^d \mid x(\mathbf{z} + \mathbf{v}) = x(\mathbf{z}), \forall \mathbf{z} \in \mathbb{Z}^d\}$ the lattice of vectors of periodicity of configuration x . This lattice Γ_x may be of any dimension below d and some cases are particularly interesting:

- When it is of dimension 0, then x does not have any vector of periodicity and is hence aperiodic.
- When it is of dimension d , x can be defined by a finite pattern. This case has been studied and partly characterized in terms of complexity classes by Jeandel and Vanier [16].
- When it is of dimension 1, then there exists some vector v such that $\Gamma_x = \mathbf{v}\mathbb{Z}$. In this case, one may talk about the direction or slope of \mathbf{v} : two vectors \mathbf{u}, \mathbf{w} have the same slope if there exists $\lambda \neq 0$ such that $\lambda \mathbf{u} = \mathbf{w}$.

In this paper, we are interested in this last case. In particular we focus on characterizing the possible sets of slopes that a subshift may realize. Having the same slope is an equivalence relation and we define a slope to be an equivalence class for this relation. We note $S(\mathbb{Z}^d)$ the sets of slopes for \mathbb{Z}^d . In dimension 2,

the usual definition of a slope is a rational $m \in \mathbb{Q} \cup \{\infty\}$ ¹ such that $y = mx$, $S(\mathbb{Z}^2)$ can thus be identified with $\mathbb{Q} \cup \{\infty\}$ as was done in [15]. The definition of slope via rationals does not scale nicely to higher dimensions however, as will be seen in Section 1.3.

We will start by studying subshifts in dimension 2:

Theorem 1 *The sets of slopes of 2-dimensional SFTs are exactly the recursively enumerable (Σ_1^0) subsets of $S(\mathbb{Z}^2)$.*

Theorem 2 *The sets of slopes of 2-dimensional sofic and effectively closed subshifts are exactly the Σ_2^0 subsets of $S(\mathbb{Z}^2)$.*

We then tackle three dimensional subshifts, where the characterization is the same for SFTs, sofic and effectively closed shifts.

Theorem 3 *The sets of slopes of 3-dimensional SFTs, sofic and effectively closed subshifts are exactly the Σ_2^0 subsets of $S(\mathbb{Z}^3)$.*

In order to do this, we introduce a new way to synchronize computations between different dimensions, partly inspired by what is done by Durand, Romashchenko and Shen [7].

The complexity gap between dimensions 2 and 3 for SFTs boils down to the fact that the subset of periodic configurations of a d -dimensional subshift along some periodicity vector may be seen as a $(d - 1)$ -dimensional subshift (see Section 2.1), which in the case of a 2-dimensional SFT means working on a 1-dimensional SFT.

We do not know whether the characterizations holds for higher dimensions since the upper bound is based on a result that is specific to \mathbb{Z}^2 [8]. However, the realization can be readily generalized to higher dimensions.

The paper is organized as follows: in Section 1 we recall definitions about subshifts and the arithmetical hierarchy, followed by the important definitions concerning periodicity in Section 1.3, in Section 2 we prove that sets of slopes of 2D and 3D SFTs and effectively closed subshifts are in the appropriate level of the hierarchy and in Section 3 we give constructions realizing sets of the arithmetical hierarchy as sets of slopes.

This article covers the results announced in [15] and [21] together with a treatment of the effective and sofic case. It also includes the full characterization allowed by [8].

1 Definitions and Properties

1.1 Subshifts and tilesets

We give here some standard definitions and facts about subshifts, one may consult [19] for more details.

¹ With ∞ representing the particular case of the vertical line.

Let Σ be a finite alphabet, a *configuration* (or tiling) is a function $c : \mathbb{Z}^d \rightarrow \Sigma$. A *pattern* is a function $p : N \rightarrow \Sigma$, where $N \subseteq \mathbb{Z}^d$ is a finite set, called the *support* of p . A pattern p *appears* in a configuration or another pattern q if there exists $\mathbf{v} \in \mathbb{Z}^d$ such that $\forall \mathbf{x} \in N, p(\mathbf{x}) = q(\mathbf{x} + \mathbf{v})$. We write then $p \sqsubseteq q$. A *subshift* is a closed, shift-invariant subset of $\Sigma^{\mathbb{Z}^d}$. With this terminology, $\Sigma^{\mathbb{Z}^d}$ is called the d -dimensional *full shift*. For a subshift X we will usually denote the shift action by σ . The full shift is a compact metric space when equipped with the distance $d(x, y) = 2^{-\min\{\|\mathbf{v}\|_\infty \mid \mathbf{v} \in \mathbb{Z}^d, x(\mathbf{v}) \neq y(\mathbf{v})\}}$ with $\|\mathbf{v}\|_\infty = \max_i |\mathbf{v}_i|$.

It is well known that subshifts may also be defined via collections of forbidden patterns. Let F be a collection of forbidden patterns, the subset X_F of $\Sigma^{\mathbb{Z}^d}$ defined by

$$X_F = \left\{ x \in \Sigma^{\mathbb{Z}^d} \mid \forall p \in F, p \not\sqsubseteq x \right\}$$

is a subshift. Any subshift may be defined via an adequate collection of forbidden patterns. A configuration of a subshift is also called a *point* of this subshift and is said to be valid with respect to the family of forbidden patterns F . A *subshift of finite type (SFT)* is a subshift which may be defined via a finite collection of forbidden patterns. In this case, remark that F being finite, one can define a subshift of finite type either by a set of forbidden or allowed patterns. Indeed, if one takes N bigger than any pattern of F , and F' the set containing all $N \times N$ squares containing a pattern of F , then $X_F = X_{F'}$. Because patterns of F' have same support, F' is uniquely defined by $\overline{F'}$ the set of $N \times N$ patterns not in F' , and so is X_F .

The subshift X_F is *effectively closed* if it is the complement of an effectively open set, that is a recursively enumerable union of basic open sets. In this case, the set F is recursively enumerable.

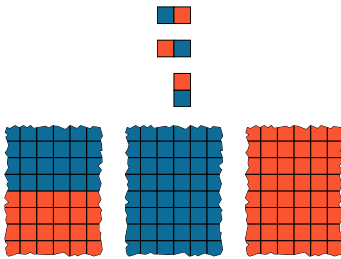


Fig. 1: A set of forbidden patterns and valid configurations associated to it.

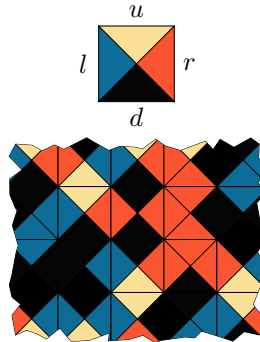


Fig. 2: A Wang tile and a portion of a valid tiling.

Wang tiles are unit squares with colored edges which may not be flipped or rotated, a *tileset* is a finite set of Wang tiles. Tiles of a tileset may be placed

side by side on the \mathbb{Z}^2 plane only when the matching borders have the same color, thus forming a tiling of the plane. The set of all tilings by some tileset is an SFT, and conversely, any SFT may be converted into an isomorphic tileset. From a computability point of view, both models are equivalent and we will use both indiscriminately. In 3D, Wang tiles can be straightforwardly generalized to Wang cubes.

A *factor* $\pi : \Sigma^{\mathbb{Z}^d} \rightarrow \Sigma'^{\mathbb{Z}^d}$ is a continuous function which commutes with the shift action. It allows to change the alphabet of a subshift by applying a local transformation to it, since such functions can always be defined locally [10]. We say that two subshifts are *topologically conjugate* (or simply *conjugate*) if there is a homeomorphism which commutes with the shift action (i.e. a bijective factor) between the two. We also use this notion to define a new type of subshift: a subshift is a *sofic subshift* if it is a factor of some SFT. Since in this paper we will talk only about three types of subshifts, we say that two subshifts are of the same type if they are both SFT, or both effectively closed or both sofic.

A subshift is *North-West-deterministic* if, for any position, and for any two colors placed above it and to its left, there exists at most one valid color at this position. Likewise, we call a subshift *West-deterministic* if it is the case with the colors to its left and top-left.

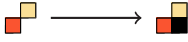


Fig. 3: NW-determinism.

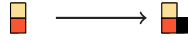


Fig. 4: W-determinism.

1.2 Arithmetical hierarchy

We give now some basic definitions used in computability theory and in particular about the arithmetical hierarchy. More details may be found in [25]. Usually the arithmetical hierarchy is seen as a classification of sets according to their logical characterization. For our purpose we use an equivalent definition in terms of computability classes and Turing machines with oracles.

An oracle for an integer set (or problem) P is an infinite word O such that $O_i = 1$ if and only if $i \in P$ in this case, the oracle is said to *accept* i . We say that a Turing machine M has O as an oracle (denoted by M^O) if M has access to an extra tape on which the word O is written. We can now define the arithmetical hierarchy:

- $\Delta_0^0 = \Sigma_0^0 = \Pi_0^0$ is the class of recursive (or computable) problems.
- Σ_n^0 is the class of recursively enumerable (RE) problems with an oracle in Π_{n-1}^0 .
- Π_n^0 the complementary of Σ_n^0 , or the class of co-recursively enumerable (coRE) problems with an oracle in Σ_{n-1}^0 .

- $\Delta_n^0 = \Sigma_n^0 \cap \Pi_n^0$ is the class of recursive (R) problems with an oracle in Π_{n-1}^0 .

In particular, Σ_1^0 is the class of recursively enumerable problems and Π_1^0 is the class of co-recursively enumerable problems.

1.3 Periodicity and aperiodicity

The notion of periodicity being central in this paper, we will define it in this section.

Definition 1 (Periodicity) A configuration c is *periodic* if there exists $\mathbf{v} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}$ such that $\forall \mathbf{x} \in \mathbb{Z}^d, c(\mathbf{x}) = c(\mathbf{x} + \mathbf{v})$, the configuration is said to be periodic of period \mathbf{v} or \mathbf{v} -periodic. If c has no period, then it is said to be *aperiodic*. A subshift is *aperiodic* if all its points are aperiodic.

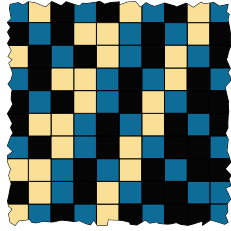


Fig. 5: Example of $(1, 2)$ -periodic configuration.

As seen in the introduction, the lattice of vectors of periodicity may be of any dimension between 0 and d and we are interested here in the case where it is 1-dimensional. In this case we can define the notion of slope of periodicity.

Definition 2 (Slope of periodicity) Let \sim be the equivalence relation such that $\mathbf{u} \sim \mathbf{v}$ if and only if there exists $\lambda \neq 0$ such that $\mathbf{u} = \lambda \mathbf{v}$. Equivalence classes of \sim are the elements of the projective plane over \mathbb{Q}^d . In this paper, we call *slope* one of its element, denoted by $[\mathbf{v}]$ for the slope of \mathbf{v} .

If all vectors of periodicity of a configuration c have the same slope s , we say that c has *slope of periodicity* (or just slope) s . We will denote $S(\mathbb{Z}^d)$ the set of all slopes of \mathbb{Z}^d .

One can now associate to a subshift X its set of slopes:

Definition 3 (Slopes of a subshift) The set of slopes of a subshift X is

$$Sl(X) = \{s \mid \exists c \in X, c \text{ has slope } s\}.$$

The set of slopes of a subshift constitutes a conjugacy invariant.

We will say that a Turing machine takes slopes as inputs, if for any vectors \mathbf{u}, \mathbf{v} given as input, the machine yields the same result for \mathbf{u} and \mathbf{v} when they have the same slope. In other words, its output only depends on the slope of the vector and not on the vector itself.

Careful reader will notice that the definition of slope changed since [15] and [21]. The reason being that the former definition did not extend nicely to dimensions greater than two. For example, with the former definition, $(0, 1, 2)$ and $(0, 1337, 1664)$ had the same slope, but $(2, 1, 0)$ and $(1664, 1337, 0)$ had not, which is not coherent with the idea that the slope represents the direction of a vector. With this new definition, this problem is solved, and the core of the proofs does not change since computations took place directly on coordinates of vectors.

2 Upper bounds

In order to give upper bounds on the complexity of deciding whether a given slope s is in a set of slopes of some subshift, we explain the classical trick that allows to work on one less dimension if we only need to work with \mathbf{v} -periodic configurations for some fixed vector \mathbf{v} .

We then proceed to give the upper bounds for 2-dimensional and 3-dimensional SFTs and effectively closed subshifts.

2.1 Reducing the dimension by fixing some periods

A classical trick, see e.g. [2] or [16] allows, for any d -dimensional SFT X and k pairwise non-colinear vectors, to construct a $(d-k)$ -dimensional SFT equivalent to the configurations periodic along all the k vectors at once. The same is true for effectively closed subshifts instead of SFTs. For sake of completeness, let us now explain this trick.

Let $\mathbf{v} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}$, σ be a shift action and X a subshift. We note $X_{\mathbf{v}} \subseteq X$ the set of \mathbf{v} -periodic configurations of X . $X_{\mathbf{v}}$ is clearly a subshift of X and is of the same type as X .

Lemma 1 *For any $\mathbf{v} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}$ non colinear to $\mathbf{e}_1, \dots, \mathbf{e}_{d-1}$, $(X_{\mathbf{v}}, \sigma|_{\mathbb{Z}^{d-1}})$ is conjugate to a $d-1$ subshift of the same type as X .*

Proof. Let $n = \langle \mathbf{e}_d, \mathbf{v} \rangle$ the scalar product of \mathbf{e}_d and \mathbf{v} , and define the projection

$$\pi_{\mathbf{v}} = \begin{cases} \Sigma^{\mathbb{Z}^d} & \rightarrow (\Sigma^n)^{\mathbb{Z}^{d-1}} \\ x & \mapsto \left((x_{(i_1, \dots, i_d)})_{0 \leq i_d < n} \right)_{(i_1, \dots, i_{d-1}) \in \mathbb{Z}^{d-1}} \end{cases}$$

Intuitively, $\pi_{\mathbf{v}}$ projects on the periodically repeated column of any \mathbf{v} -periodic configuration. The restriction of $\pi_{\mathbf{v}}$ to $X_{\mathbf{v}}$ is a bijection. Figure 6

shows how the forbidden patterns of X are transformed into forbidden patterns of $\pi_{\mathbf{v}}(X)$. The type of the subshift is clearly conserved via this transformation. \square

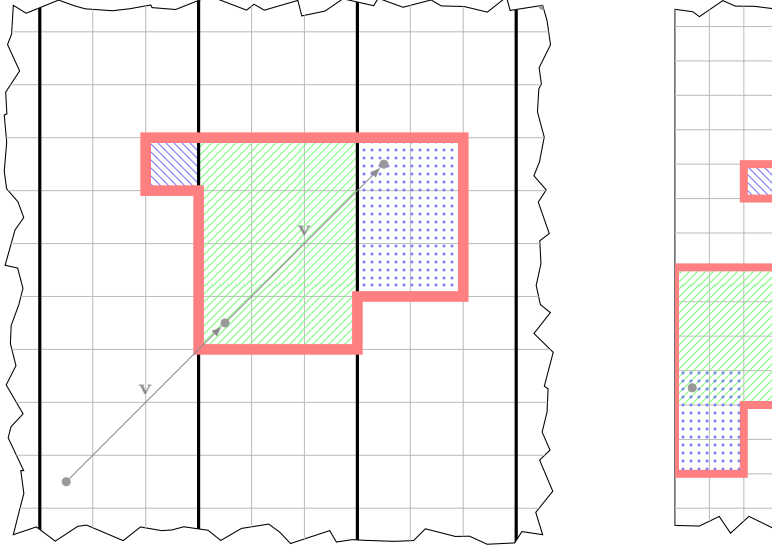


Fig. 6: On the left, the support of a forbidden pattern in a 2-dimensional \mathbf{v} -periodic configuration and on the right how this support is transformed for the 1-dimensional equivalent subshift.

Repeated applications of the previous lemma lead to the following corollary:

Corollary 1 *Let $\mathbf{v}_1, \dots, \mathbf{v}_k$ be pairwise non colinear vectors of \mathbb{Z}^d and $\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}$ be generating vectors non colinear with any of the $(\mathbf{v}_i)_{1 \leq i \leq k}$.
 $\left(\bigcap X_{\mathbf{v}_i}, \langle \sigma_{\mathbf{e}_{i_1}}, \dots, \sigma_{\mathbf{e}_{i_{d-k}}} \rangle \right)$ is conjugate to a $(d-k)$ -dimensional subshift of the same type as X .*

2.2 Slopes of SFTs of dimension 2

In a first time, let us remark the following.

Lemma 2 *Let Y be a one-dimensional SFT. Then it is decidable to check if Y contains an aperiodic configuration.*

Proof. Y being one-dimensional, it could be represented by a finite graph G . Let k be an integer bigger than any of its forbidden patterns. The vertices of G are all possible k -pattern on Σ and (u, v) is an edge of G if and only if uv is

a valid pattern of size $2k$ for Y . Using this graph, it is easy to find if Y has an aperiodic configuration. Indeed, it is the case if and only if G has two distinct cycle such that one is accessible from the other, which is decidable since G is finite. \square

Lemma 3 *Let X be a two-dimensional SFT. Then $Sl(X)$ is a Σ_1^0 set.*

Proof. We have $s \in Sl(X)$ if and only if there exists a configuration $c \in X$ and a vector \mathbf{v} with slope s such that c is 1-periodic along \mathbf{v} . Let us call $Y_{\mathbf{v}}$ the one-dimensional SFT to which $X_{\mathbf{v}}$ is conjugated by Lemma 1. Because $X_{\mathbf{v}}$ is the set of all \mathbf{v} -periodic configurations of X , there exist a 1-periodic configuration along \mathbf{v} in X if and only if $Y_{\mathbf{v}}$ is nonempty and contains an aperiodic configuration, which is decidable by Lemma 2.

We can use this to build a Σ_1^0 algorithm to check if $s \in Sl(X)$: we enumerate all \mathbf{v} such that $[\mathbf{v}] = s$ and halt if $Y_{\mathbf{v}}$ has an aperiodic configuration i.e. X has a 1-periodic configuration with slope s . \square

2.3 Slopes of 3-dimensional SFTs, sofic and effectively closed subshifts of dimensions 2 and 3

For three dimensional SFTs, using the same trick as before leads to a two dimensional SFT where most problems are undecidable (Lemma 2 does not hold for two dimensional SFTs). We will therefore directly treat the effective case.

In order to obtain an upper bound in the case of 2 and 3-dimensional effective subshifts, we will need the following fact:

Theorem 4 (Grandjean, Hellouin, Vanier [8]) *Let X be a one or two dimensional effectively closed subshift. Then the set of aperiodic configurations of X is a Π_1^0 set.*

Remark 1 In [8] the theorem is stated for 2D SFTs, however the proof actually works for 2D effectively closed subshifts as the function g the authors uncovered is the same for all subshifts be they SFTs, effective or not. A simplification of the whole argument developed also leads to the same result in dimension 1.

Lemma 4 *Let X be a 2 (resp. 3-dimensional) effectively closed subshift. Then $Sl(X)$ is a Σ_2^0 set.*

Proof. The proof of this lemma is analogous to the proof of Lemma 3, except that now, checking whether $Y_{\mathbf{v}}$ contains an aperiodic configuration is no longer computable but Π_1^0 according to Theorem 4. \square

3 Realizations

In this section we will realize the computability obstructions obtained in the previous section:

- We will start by giving a realization of Σ_1^0 sets of slopes by 2D SFTs.
- We then realize Σ_2^0 sets as sets of slopes by 2D effectively closed subshifts.
- We finish by constructing all Σ_2^0 sets as sets of slopes of 3D SFTs.

These results together with the upper bounds of the previous section prove Theorems 1, 2 and 3.

3.1 Two-dimensional subshift of finite type

In dimension two, the problem of checking whether some slope $s \in Sl(X)$ turns out to be Σ_1^0 -hard.

Lemma 5 *Let $R \subseteq S(\mathbb{Z}^2)$ be a Σ_1^0 set. Then there exist a 2D SFT X such that $R = Sl(X)$.*

Proof. We use for this proof techniques similar to [14, 16]. We will construct for each Turing machine M , corresponding to the recursively enumerable set R , an SFT X whose slopes are exactly the ones accepted by M . We assume that M takes as input slopes in binary, as defined in Section 1.3.

We will first build an SFT X that has as slopes $\{(p, q) \in R \mid p > q > 0\}$. The other cases are treated in the same way and the final SFT is the disjoint union of the SFT treating each case. The special cases $p = 0$, $q = 0$, and $p = q$ will be shortly discussed later on.

For the particular case where $p > q > 0$ we want to enforce the fact that when a configuration has exactly one direction of periodicity, this direction of periodicity has to be accepted by the Turing machine M . The SFT X_M will enforce the skeleton described in Fig. 7, where each square encodes the computation by M proving that the slope $[(p, q)]$ is accepted. For this, we need the size of the square to be arbitrarily large independently of $[(p, q)]$, so that the computation of M has enough time to accept. This skeleton in itself could be biperiodic, we will then color the background of each square to ensure the existence of configurations with only one direction of periodicity.

In order to enforce this skeleton, we will use several layers (or components), each of them having their own aim, and impose some constraints on how the layers may combine. We give here $X_M = C \times R \times W \times S \times P \times T_M \times A$ where:

- C will allow us to make the rows and columns,
- R will force squares to appear,
- W will force the periodicity vector and write the input for the Turing machine,
- S will force the aperiodic background of the squares to be the same in all squares,
- P will reduce the size of the input,
- T_M will code the Turing machine M in the squares,
- A will allow slopes of unique periodicity to appear.

We will now proceed to the details of the proof, by giving each component and explaining what it enforces.

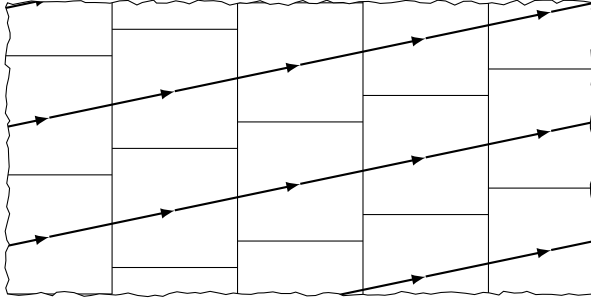


Fig. 7: Skeleton of the configurations of X_M : when the configuration is periodic, the squares appear and each of them is the shifted version of its lower left neighbor. Inside the squares we will encode the Turing machine.

Component C. The first component is made of an **East-deterministic** aperiodic SFT. We will call all of its alphabet white cells (the white background of Fig. 7), and we add two sets of cells: the horizontal breaking cells $\{\blacksquare\}$ and the vertical breaking cells $\{\blacktriangleright, \blacktriangleleft, \blacksquare, \square\}$ (the horizontal and vertical lines of Fig. 7). The rules are simple:

- on the left of a \blacksquare there can only be a \blacksquare or a \blacktriangleright ,
- on the right of a \blacksquare there can only be a \blacksquare or a \blacktriangleleft ,
- above and below a \blacksquare , there can only be a white,
- above a \blacktriangleright can only be a \blacksquare ,
- above a \blacksquare can only be a \blacktriangleright or a \blacksquare ,
- above a \blacktriangleleft can only be a \square ,
- above a \square can only be a \blacktriangleleft or a \square .

To put it in a nutshell, it means that horizontal breaking cells forms rows that can only be broken by vertical breaking cells, and vertical breaking cells can only form columns that cannot be broken.

In a periodic configuration, we cannot have a quarter of plane filled with white (it will be an aperiodic configuration). As a consequence, periodic configurations at this stage are necessarily formed by a white background broken *infinitely many times* by horizontal or vertical breaking cells.

One more rule we add is that the rules on white cells “jump” over the black cells. That is to say if we remove a black row, then the white cells have to glue themselves together correctly. The valid periodic configurations at this stage are represented on Fig. 8.

Component R. The next component will force the apparition of squares between two columns of vertical breaking cells and prevent several infinite rows of horizontal breaking cells to appear. This layer is made with the alphabet $\{\blacksquare, \blacksquare, \blacktriangleright, \blacksquare, \square, \blacktriangleleft, \square\}$, the rules applied on this layer are given by Wang tiles. We superimpose the alphabets with following rules:

- \blacksquare can only be superimposed to \blacksquare, \square ,
- \blacktriangleright can only be superimposed to \blacksquare ,

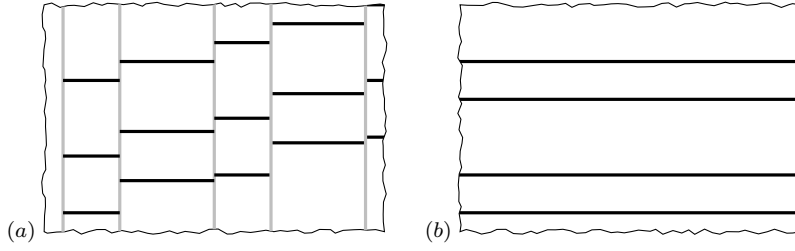


Fig. 8: Valid periodic configurations are formed of columns of vertical breaking cells (a) or of rows of horizontal breaking cells (b). Between two columns of vertical breaking cells there can be rows of horizontal breaking cells.

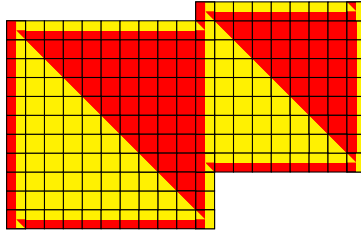


Fig. 9: Component R forces squares.

-  goes on , and  goes on ,
- , ,  are superimposed to the white cells.

Fig. 9 shows how this component R forces rows of black cells to appear between two gray columns. The distance between these black rows is exactly the distance between the gray columns thus black rows and gray columns form squares. At this stage the valid periodic configurations cannot be formed of only rows of black cells anymore.

Component W . What this component does is that it synchronises the offsets between squares of two neighboring columns, and forces all columns to be at equal distance of their two neighboring columns, for all of them. As a side effect, it also writes the offset between two squares (which we call q) in each square. In order to do that, what we do is that we prolongate the black rows of each column into their direct neighbors with two new layers, one for the left and one for the right. The end of the black row then sends a diagonal signal which changes its direction when it collides with the projected lines of the neighbors and its collision with the column has to coincide with the projection of the other column. Fig. 10.a shows how this mechanism works. The collision of the signal sent on the right extremity of the black lines marks the end of the input q on each square. We add two other sublayers to make the white rows of same width. The first one sends a signal from the left extremity of a black line which has to meet the next

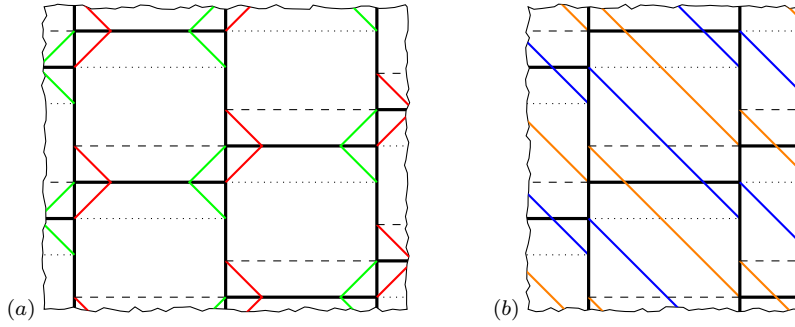


Fig. 10: The dotted row (resp. dashed) corresponds to the prolongation on the right (resp. left) of the black cells. In (a) the signals sent from the extremities of the rows forming the square forces the offset between rectangles of three neighboring columns to be exactly the same for any of them. In (b) the signals sent from the extremities force the distance between columns to be identical.

column at the exact point of the extension of the square. The second one does the same for the right extremity. Fig. 10.b shows these signals.

Component S . This component is meant to synchronize the aperiodic backgrounds of all the squares. In order to do that, we only need to transmit the first column after a vertical breaking column since our initial aperiodic SFT is East-deterministic.

In order to do that, we take the alphabet $\{\boxminus, \boxplus, \boxtimes, \boxdot\}$, with the following rules:

- on the right, above and below a \boxplus there can only be a \boxplus or a \boxtimes .
- on the left of a \boxtimes we necessarily have a \boxtimes and the south western neighbor of a \boxtimes , if the cell is a white, is a \boxtimes or a \boxdot ,
- the lower left white cell of a square is necessarily a \boxdot . The rules on \boxdot is that there can only be a \boxdot or a \boxplus on a white cell to its right,
- the vertical/horizontal breaking cells have necessarily a \boxminus on them.

The configuration obtained inside a square is shown on Fig. 11. We add a sublayer that is a copy of the white cells with the rules that the cells of this component on the right of this column are identical to the white ones on component C and that this copy is transmitted to the cell pointed by the arrow. Then with the property that the black cells continue the rules on the whites, the whole aperiodic background between two vertical breaking columns is exactly the same but shifted by the offset.

Component P . Now each square contains two data: its size (p) and the offset to the next square q , both in unary. We will pass them as input to the Turing machine after some transformation.

The idea is to transform the unary input (p, q) into a smaller binary one (p', q') where $\gcd(p', q')$ is not a multiple of two. Doing that is fairly easy: we first need to convert the input in binary; this can be done by the iteration

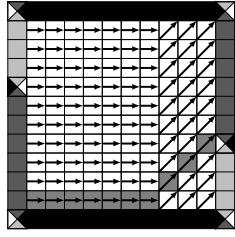


Fig. 11: Cells allowing to transmit the aperiodic background.

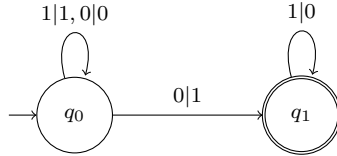


Fig. 12: A transducer transforming n in binary into $n + 1$.

of the transducer of Fig. 12: starting from $000 \dots 00$ we obtain the binary representation of p (least significant bit on the rightmost part) in p iterations of the transducer. Then we strip the binary representation of p and q of their common last zeroes.

Component T_M . This layer implements the Turing machine M as usually done in SFTs, with its input already computed by layer P . Note that the Turing machine has to halt for the tiling to be valid.

Component A . This layer is made of only two colors, yellow and blue. It will be superimposed to white cells and to the \blacksquare of the vertical breaking lines of component C only. The rules are that two neighboring cells (horizontally and vertically) have the same color. It is easy to see that the color is uniform inside a square and that it spreads to the upper right and lower left neighboring squares. Thus the squares along the direction of periodicity have the same color.

We now prove that the preceding construction works.

3.1.1 Any slope is an accepted input of M

Let $[(p, q)] \in Sl(X)$ be a slope of periodicity of X , with $p > q > 0$. By construction, the configuration has to be formed of squares of identical size with constant offset (components C , R , W). Their aperiodic background has to be the same on each column (component S), so that in fact the configuration is periodic along direction (m, n) where m and n denote respectively the width and offset of the configuration. As a consequence, the configuration has slope $[(m, n)] = [(p, q)]$ because $(m, n) = 2^k k'(p, q)$ for some k, k' with k' odd.

Now the Turing Machine on each square has $(k'q, k'p)$ as an input and halts. Hence the slope $[(k'p, k'q)] = [(p, q)]$ is accepted by the machine, so $[(p, q)] \in R$, which proves $Sl(X) \subseteq R \cap \{[(p, q)] \mid p > q > 0\}$.

3.1.2 Any accepted input of M is a slope of some configuration

Let $[(p, q)] \in R$ be an accepted input of M with $p > q > 0$. There exists a time t and a space s such that M accepts $[(p, q)]$ in time t and space s , with $s \leq t$. Take $(m, n) = 2^{\lceil \log t \rceil} (p, q) \geq (t, s)$. Now the $m \times m$ square is big enough for the computation on input (p, q) to succeed. Hence there is a configuration of period (m, n) and component A allows us to make the direction of periodicity unique by dividing the plane into two colors, half a plane yellow and half a plane blue. Hence $R \cap \{[(p, q)] \mid p > q > 0\} \subseteq Sl(X)$.

This finishes the proof for the case $p > q > 0$.

3.1.3 Other cases

The cases where $q > p > 0$, $-p > q > 0$, or $q > -p > 0$ are treated in a very similar way: rotating the SFT we just constructed and changing the way the input is written on the tape (to invert the inputs, or add a minus sign) is enough. However the remaining cases ($p = \pm q, p = 0, q = 0$) need special treatment.

For these cases, the construction above does not work, by that we mean that just rotating it and modifying slightly the Turing machine of component T_M won't do the trick. However it is actually simpler. We now make squares facing one another, obtaining a regular grid. This requires less cells for component C and no component W . Then according to the case, components C, S and A are modified as follows:

- for $p = q$, S just transmits diagonally the cells. In component A , the color is synchronized from the top right corner to the next square at the north east. The case $p = -q$ is similar.
- for $q = 0$, S transmits horizontally, and the colors of component A are synchronized with the square on the right. The configuration can only be horizontally periodic if the Turing machine accepts it, this is the only way it can be periodic.
- for $p = 0$, C has, instead of an east deterministic SFT, a north deterministic one. Components S and A are modified accordingly. The configuration can only be vertically periodic if the Turing machine accepts it and this is the only way it can be periodic.

□

Remark 2 Note that the special cases $p = 0, q = 0, q = p$ and $q = -p$ could have been treated by just adding a particular configuration for each of them if they were in R , but by doing so one would have lost the uniformity of the construction.

Recall that Lemma 3 proved that any set of slope of a 2D SFT is in Σ_1^0 , together with Lemma 5, this proves Theorem 1.

3.2 Two-dimensional sofic subshifts

If instead of building a subshift of finite type we allow ourselves to build an effectively closed subshift, we are able to use the extra computational power to encode a Π_1 oracle in the subshift in addition to the previous construction, leading to realizations of Σ_2^0 sets. This can be pushed even further by using the fact that effectively closed subshifts of dimension 1 are subactions of sofic subshifts of dimension 2.

To prove the next lemma, we will need to use the following fact due independently to Aubrun and Sablik [1] and Durand Romashchenko and Shen [6], both improving a result of Hochman [11]:

Theorem 5 (From [1] or [6]) *For any one-dimensional effectively closed subshift X , there exist a two-dimensional sofic subshift X' such that*

$$X' = \left\{ \begin{array}{c|c} \vdots & \\ \vdots & x \\ x & \\ x & \\ \vdots & \end{array} \middle| x \in X \right\}.$$

Lemma 6 *Let $R \subseteq S(\mathbb{Z}^2)$ be a Σ_2^0 set. Then there exist a 2D sofic subshift X such that $R = Sl(X)$.*

Proof. By definition of Σ_2^0 , there exists a Turing machine M and a Π_1^0 oracle O such that $R = \{[(p, q)] \mid M \text{ with oracle } O \text{ accepts the slope } [(p, q)]\}$. The SFT X such that $Sl(X) = R$ will be very similar to the one built in Lemma 5, but with some small modifications to include the oracle in it.

We give the proof only for the case $p > q > 0$ since the other cases can be treated as in the previous theorem.

Let $X_{MO} = C \times R \times W \times S \times P \times A \times T_O \times T_{MO}$, with C, R, W, S, P and A as defined before. With only these six layers the 1-periodic configurations of X_{MO} are formed of 1-periodic squares filled with aperiodic background. These squares also have their offset encoded in binary into them. Instead of directly encoding the Turing machine as done previously, we first create a new layer T_O , which will contain the computation of the oracle.

Let us assume that in every square the layer T_O always contains the same 0/1 word repeated vertically: this word will correspond to the beginning of the oracle tape. We encode in T_{MO} the Turing machine M whose tape is placed horizontally and whose time flows vertically. We also give it access to a second tape corresponding to the word of layer T_O .

Now we want the layer T_O to actually encode the oracle tape and this is where we need a sofic subshift instead of an SFT. We first build a one-dimensional effectively closed subshift that builds the oracle tape, and then

apply Theorem 5 to make a two-dimensional sofic subshift that has the same configurations as the effective one.

Because O is a Π_1^0 oracle, there exists a Turing machine M_2 such that if O contains a 1 at position i then M_2 does not halt on i . Let us define the subshift $Y = \{\dots w\#w\#w\#\dots \mid \forall i, w_i = 1 \Rightarrow M_2 \text{ does not halt on input } i\}$. It is effective because it is defined by the following recursively enumerable set of forbidden words:

$$F = \{w \mid \exists i, w_i = 1 \text{ and } M_2 \text{ halts on input } i\}.$$

Indeed, $w \notin F \Leftrightarrow [\forall i, w_i = 1 \Rightarrow M_2(i) \uparrow] \Leftrightarrow \dots w\#w\#w\#\dots \in Y$. Thanks to Theorem 5, there exists a two-dimensional sofic subshift Y' such that

$$Y' = \left\{ \begin{array}{c} \vdots \\ \dots w \# w \# w \# \dots \\ \dots w \# w \# w \# \dots \\ \dots w \# w \# w \# \dots \\ \vdots \end{array} \middle| \forall i, w_i = 1 \Rightarrow M_2 \text{ does not halt on input } i \right\}.$$

The layer T_O is build from this sofic subshift Y' and we add the constraint that the columns of $\#$ have to be superimposed to the vertical breaking lines of layer C . So, layer T_{M^O} contains finite words where the ones can be trusted but not the zeroes, this will however not matter as the only cases we are interested in are the cases when M halts with oracle w , and we can force M to recheck all zeroes, if it halts for all needed zeroes, then it will be able to carry its normal computation and halt, otherwise it won't so the only possible w that will appear in 1-periodic configurations are the prefixes of O .

Now let us prove that this new construction achieves what we want. Let $[(p, q)] \in Sl(X_{M^O})$ be a slope of periodicity of X_{M^O} , with $p > q > 0$. The construction being the same as in Lemma 5, we still have a 1-periodic configuration with squares and the slope encoded on the border of the squares. Since the squares are finite the Turing machine M encoded into them halts, and it uses a finite part of the oracle tape O . Therefore, the input is accepted by M^O : $[(p, q)] \in R$.

Conversely, let $[(p, q)] \in R$ be an accepted input of M^O . with $p > q > 0$. By definition there is a finite execution of M that accepts the slope of (p, q) . This machine uses a finite portion of the oracle tape. Therefore, there exists a big enough square to contain all the computation of M that we can use to build a 1-periodic configuration of X_{M^O} : $[(p, q)] \in Sl(X_{M^O})$. \square

Together with Lemma 4, this proves Theorem 2.

3.3 Three-dimensional subshifts of finite type

Adding a third dimension will allow us to match the Σ_2^0 bound with SFTs.

Lemma 7 *Let $R \subseteq S(\mathbb{Z}^3)$ be a Σ_2^0 set. Then there exists a 3D SFT X such that $Sl(X) = R$.*



Proof. The global idea of the proof is similar to Lemma 5, but instead of squares we will be making cubes with computation in them. The extra dimension will provide a way of modifying the construction to embed in it a Π_1^0 oracle without using an effective subshift, showing a Σ_2^0 lower bound as in Lemma 6.



Let M be a Turing machine accepting rationals from R with an oracle $O \in \Pi_1^0$.

As done in Lemma 6 we only explain the case for slopes $[(p, q, r)]$, with $p > q > r > 0$, the others are symmetric or quite similar and it suffices to take the disjoint union of the obtained SFTs to get the full characterization. We build an SFT X_{M^O} such that its 1-periodic configuration are formed of large cubes, shifted with an offset to allow periodicity along some slope. Then we encode M inside all the cubes, and give to it the slope as input. The machine halts (i.e the slope is in R) implies that the cubes are of finite size. Which means that the configuration is 1-periodic only when the slope actually corresponds to some element of R . As before, we build this SFT using several layers: $X_{M^O} = B_{yz} \times B_{xz} \times B_{xy} \times C \times W \times P \times S \times A \times T_O \times T_{M^O}$, with:

- $B_{yz} \times B_{xz} \times B_{xy} \times C$ essentially does the same as component C of Lemma 5: it forces the apparition of large cubes separated by chunks of an aperiodic tiling,
- W synchronises the offsets between the cubes and write the input of the Turing Machine in the border of the cubes,
- S forces the aperiodic background of the squares to be the same in all squares,
- P reduces the size of the input,
- A allows slopes of unique periodicity to appear,
- T_{M^O} encodes the “ Σ_2^0 ” Turing machine M in the cubes,
- T_O encodes the Π_1^0 oracle O that is used by M .

Aperiodic background. We first need an aperiodic background in order to ensure that there is no other directions of periodicity that the one we create later on. We even make this background West-deterministic on planes (xz) and (yz) , since some layers will need that deterministic property to work. For that we cross two 2D West-deterministic aperiodic backgrounds: the set of aperiodic cubes are the sets of cubes of the form shown in Fig. 13. We also impose that all parallel planes are identical (Fig. 14). The 2D aperiodic tiling is again the NW-deterministic from Kari [17] modified to be West-deterministic. With such a superposition, one can easily show that the resulting 3D tiling is aperiodic.

Component B_{yz} . The first layer is made with two types of cubes: a white cube () which is a meta-cube that corresponds to any cube of the aperiodic background and a black cube () which will serve to break the aperiodicity brought by the white cubes. The rules of this layer are:

- In coordinates $z + 1$ and $z - 1$ of , only a  can appear.

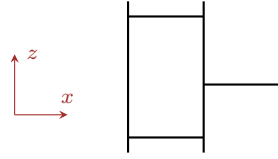


Fig. 15: Projection on the (xz) plane of a valid configuration with layers B_{yz} and B_{xz} .











-  can only be on 
-  can only be on  and  only on 
- ,  and  can only be on white tiles 

Fig. 9 shows how this layer forces squares, and therefore cube, to appear.


Component W . This layer uses signals to synchronize the offsets of different cubes, and to force cubes to have the same size. It also writes the different offsets in the border of the cubes. This is essentially the same as layer W of Lemma 5 but duplicated on (xy) and xz planes to take care of the three possible offsets. Fig. 10 shows these offsets in the 2D case.

Component P . This layer reduces the size of the input, in order to allow us to construct valid configurations as large as we want for the same input (p', q', r') . Starting from an unary input (p, q, r) , this layer writes into cubes what the input of the Turing machine will be: (p', q', r') , with $(p, q, r) = 2^k(p', q', r')$, and $\gcd(p', q', r')$ not divisible by 2. We use the same method as in Lemma 5 by using finite transducers and encoding them into the tiling.

Component S . Aperiodic backgrounds of different “slices” may be different (“slices” are the thick planes in the (yz) plane). They must be synchronized in order to ensure the existence of a periodic configuration along (p, q, r) . We already know how to do this synchronization in 2D, and a small trick will allow us to do the same in 3D. We create two layers of 2D arrows synchronizing independently 2D backgrounds. One in the (xy) plane, that are repeated along z (*front* arrows) and the other in (xz) and repeated along y (*top* arrow). We then create our real 3D layer using these two 2D layers, with 3D arrows:



The superimposition of two 2D arrows gives directly which 3D arrow is on each tile (see Fig. 16). Like in 2D we impose that the background is the same at the beginning and at the end of the gray arrow. Thanks to the double West-periodicity of the background, this ensures that the background has a periodicity vector of (p, q, r) in valid configurations.

Component A . This layer forces the apparition of 1-periodic configurations. Using two cubes (yellow and blue), superimposed only with the white cubes inside of the big cubes and black cubes  of layer B_{yz} . We impose that yellow or blue neighbors have the same color. It is easy to see that the color

front	top	3D

Fig. 16: Rules for layer S (transmission in 3D).

is uniform inside a cube and spread to two opposite corners of cubes. Thus all the cubes along (p, q, r) have the same color and there exists at least one 1-periodic configuration.

Component T_{M^O} . This layer encodes the Turing machine M in the tiling. The idea for the Turing machine is very similar to Lemma 6, it will have access to an extra tape corresponding to the Π_1^0 oracle tape. Here again, we will encode M with an additional read-only arbitrary tape, and the next layer will ensure that the content of the tape is valid for O . The tape is a line along axis y duplicated along axes x and z (see Fig. 17). We add the two rules:

1. Inside a cube, a number at position x is the same as the number at position $x - 1$ and a number at position z must be equal to the number at position $z - 1$.
2. The first line of the tape is transmitted through black cubes like the aperiodic background.

The first rule duplicates the first line everywhere inside a cube, and the second one ensures that the same R_O tape is duplicated along the direction of periodicity.

Then, we encode M in the (xz) plane using the usual encoding of Turing machines in tilings. Let us say that the time is along the z axis and the working tape along x . In order to access the entire R_O tape, we add the spacial dimension y to the TM encoding: while doing a transition, the machine can move its head along the y axis and read the value of the R_O tape in it; rule 1 above prevents M to modify this extra tape.

Component T_O . This layer is the core of this proof, and it is where 3D actually comes to play: in the thick aperiodic (yz) planes we will compute the Π_1^0 oracle by encoding an infinite computation that checks simultaneously all possible inputs of M^O , the Turing machine checking the Π_1^0 oracle O (M^O halts if and only if there is a wrong 1 in the portion of R_O written in all the cubes). Thanks to the 3rd dimension we have, we can do this using only SFT rules.

The key idea of this layer is the use of the previously constructed cubes as *macro-tiles* in order to encode computations of M^O . Each cube will thus represent one tile and the thick aperiodic planes will contain, more sparsely, another 2D tiling. See Fig. 19 to see how the cubes store this macro-tileset. For this macro-tileset, we may use a construction of Myers [22] which

modifies Robinson's aperiodic tileset in order to synchronize the input tapes on all of the partial computations. So each of our cubes contains/represents one tile of Myer's tileset, and the thick aperiodic planes thus also contain a Myers tiling checking some input that for the moment is not synchronized with the oracle written inside these cubes.

We now have a valid macro-tiling for the cubes if and only if the machine M^O never halts on R_O .

The one remaining thing to do is to explain the R_O tape that M^O accesses. For M^O to be able to access it any time, we will store it inside the large cubes. We add to the set of numbered tiles the same tiles, but in red, representing the head of the M^O Turing machine on the tape R_O . We impose that there is only one red number in every large cube (see Fig. 18). The red tile of a cube must be synchronized with the cell of the oracle R_O currently contained in the Myers tile. Every time a new partial computation is started in the macro-tiles, the red tile must be placed at the beginning of R_O , whenever the macro-tile moves the head to the right/left, the red tile must also be moved, if the red tile reaches the border of the cube, in which case it reaches a special state of non-synchronization, since the beginning has already been synchronized.

To do that, we must allow two new transitions. These new transitions do not change the state of the working tape, thus we only move to the next time along z . But in the new position, the red-marked cell in the large cube must have changed. To do this, we again use signals between the bottom-cube (previous state), the cell doing the transition and the upper-cube (next state), see Fig. 20.

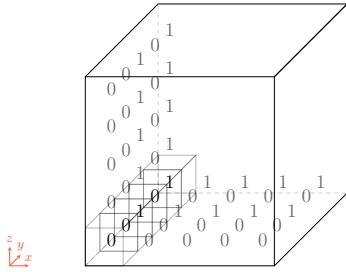


Fig. 17: Tape R_O of the oracle in a cube.

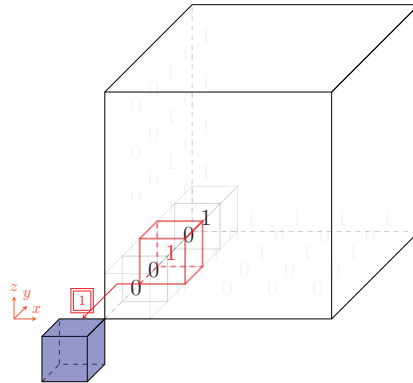


Fig. 18: The red tiles on R_O and the transmission of its value.

Now we prove that this construction does what we claim, finishing the proof of Theorem 3.

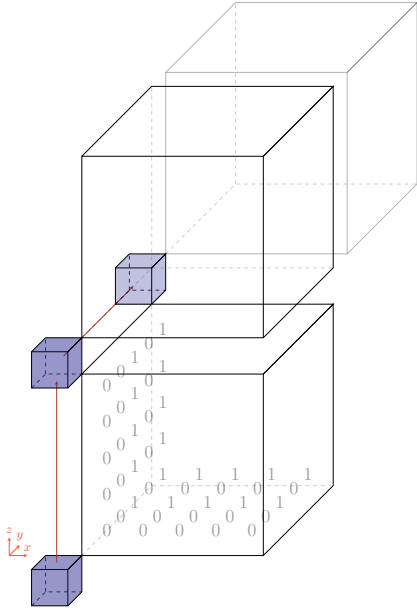


Fig. 19: Meta-tiles of large cubes, with the adjacency rules represented by the arrows. Myers' tiles are placed in the darker cubes, in the (yz) plane.

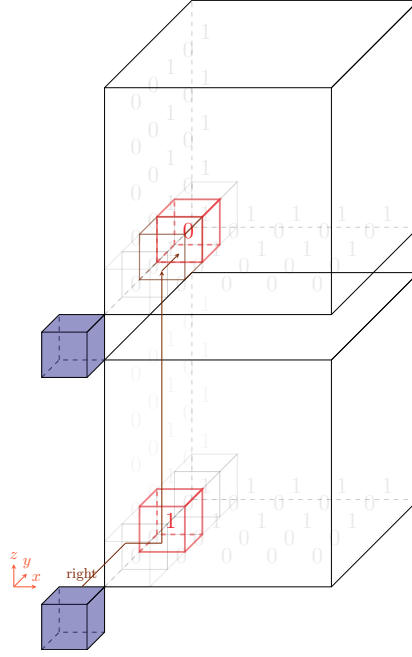


Fig. 20: Moving the red cube when the head of M^O moves.

3.3.1 Any slope is an accepted input of M^O

Let $[(p, q, r)]$ be a slope of periodicity of some configuration of X_M . By construction every periodic configuration along this slope is formed with cubes of the same size p , shifted with the same offset (q, r) . Every cube has the same content, which corresponds to an execution of M . Cubes being of finite size, every execution is a halting execution of M . Let's take $(p, q, r) = 2^k(p', q', r')$, with p', q', r' odds. Thanks to the layer P, the input of M is (p', q', r') , then M accepts the slope of (p', r', q') , which is the same as (p, q, r) .

3.3.2 Any accepted input of M^O is a slope of some configuration

If M accepts the slope of (p, q, r) , there exists a time t , a space a on the working tape and b on the oracle tape, in which the machine M halts. We can assume $t \geq a \geq b$ without loss of generality. Take $m = 2^{\lceil \log t \rceil} p \geq t \geq a \geq b$. Then, the $m \times m$ cube is big enough to contain the computation of M . The configuration formed by cubes of size m and of offset $(n, o) = 2^{\lceil \log t \rceil} (q, r)$ has slope $[(m, n, o)] = [(p, q, r)]$. \square

This lemma combined with the upper bound of Lemma 4 gives Theorem 3. This realization of Σ_2^0 sets as set of slope can be easily generalized for higher dimensions than three. Indeed, layers that do not contain any computation can be easily extended with the same tricks used to go from dimension two to three. Layers with computation can just use the three first dimensions, and we obtain the desired SFT. However note that because of the alternant nature of the definition of the arithmetical hierarchy, extra dimensions from then on do not add any computational power.

Acknowledgements The authors would like to particularly thank Ville Salo for some very useful remarks on a previous version of this paper that led to a better exposition and to some corrections.

References

1. Aubrun, N., Sablik, M.: Simulation of effective subshifts by two-dimensional subshifts of finite type. *Acta Applicandae Mathematicae* **126**(1), 35–63 (2013). DOI 10.1007/s10440-013-9808-5
2. Ballier, A., Jeandel, E.: Structuring multi-dimensional subshifts. CoRR **abs/1309.6289** (2013). URL <http://arxiv.org/abs/1309.6289>
3. Berger, R.: The Undecidability of the Domino Problem. Ph.D. thesis, Harvard University (1964)
4. Berger, R.: The Undecidability of the Domino Problem. No. 66 in *Memoirs of the American Mathematical Society*. The American Mathematical Society (1966)
5. Culik II, K., Kari, J.: An aperiodic set of Wang cubes. *Journal of Universal Computer Science* **1**(10), 675–686 (1995)
6. Durand, B., Romashchenko, A., Shen, A.: Effective closed subshifts in 1d can be implemented in 2d. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **6300 LNCS**, 208–226 (2010). DOI 10.1007/978-3-642-15025-8_12
7. Durand, B., Romashchenko, A., Shen, A.: Fixed-point tile sets and their applications. *Journal of Computer and System Sciences* **78**(3), 731–764 (2012). DOI 10.1016/j.jcss.2011.11.001
8. Grandjean, A., de Menibus, B.H., Vanier, P.: Aperiodic points in \mathbb{Z}^2 -subshifts. In: *Automata, Languages, and Programming - 45th International Colloquium, ICALP 2018, Prague, Czech Republic, July 9-13, 2018, Proceedings* (2018)
9. Gurevich, Y., Koryakov, I.: Remarks on Berger’s paper on the domino problem. *Siberian Math. Journal* pp. 319–320 (1972)
10. Hedlund, G.A.: Endomorphisms and automorphisms of the shift dynamical system. *Mathematical systems theory* **3**(4), 320–375 (1969). DOI 10.1007/BF01691062. URL <https://doi.org/10.1007/BF01691062>
11. Hochman, M.: On the dynamics and recursive properties of multidimensional symbolic systems. *Inventiones mathematicae* **176**(1), 131 (2008). DOI 10.1007/s00222-008-0161-7. URL <https://doi.org/10.1007/s00222-008-0161-7>
12. Hochman, M., Meyerovitch, T.: A characterization of the entropies of multidimensional shifts of finite type. *Annals of Mathematics* **171**(3), 2011–2038 (2010). DOI 10.4007/annals.2010.171.2011
13. Jeandel, E., Rao, M.: An aperiodic set of 11 wang tiles. CoRR **abs/1506.06492** (2015). URL <http://arxiv.org/abs/1506.06492>
14. Jeandel, E., Vanier, P.: Periodicity in Tilings. In: *Developments in Language Theory (DLT)* (2010)
15. Jeandel, E., Vanier, P.: Slopes of tilings. In: J. Kari (ed.) *JAC*, pp. 145–155. Turku Center for Computer Science (2010)

16. Jeandel, E., Vanier, P.: Characterizations of periods of multi-dimensional shifts. *Ergodic Theory and Dynamical Systems* **35**, 431–460 (2015). DOI 10.1017/etds.2013.60. URL http://journals.cambridge.org/article_S0143385713000606
17. Kari, J.: The Nilpotency Problem of One-Dimensional Cellular Automata. *SIAM Journal on Computing* **21**(3), 571–586 (1992)
18. Kari, J.: A small aperiodic set of Wang tiles. *Discrete Mathematics* **160**, 259–264 (1996)
19. Lind, D.A., Marcus, B.: *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, New York, NY, USA (1995)
20. Meyerovitch, T.: Growth-type invariants for \mathbb{Z}^d subshifts of finite type and arithmetical classes of real numbers. *Inventiones Mathematicae* **184**(3) (2010). DOI 10.1007/s00222-010-0296-1
21. Moutot, E., Vanier, P.: Slopes of 3-dimensional subshifts of finite type. In: F.V. Fomin, V.V. Podolskii (eds.) *Computer Science - Theory and Applications - 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, June 6-10, 2018, Proceedings, Lecture Notes in Computer Science*, vol. 10846, pp. 257–268. Springer (2018). DOI 10.1007/978-3-319-90530-3_22
22. Myers, D.: Non Recursive Tilings of the Plane II. *Journal of Symbolic Logic* **39**(2), 286–294 (1974)
23. Ollinger, N.: Two-by-Two Substitution Systems and the Undecidability of the Domino Problem. In: *CiE 2008*, no. 5028 in *Lecture Notes in Computer Science*, pp. 476–485 (2008)
24. Robinson, R.M.: Undecidability and Nonperiodicity for Tilings of the Plane. *Inventiones Mathematicae* **12**(3), 177–209 (1971). DOI 10.1007/BF01418780
25. Rogers Jr., H.: *Theory of Recursive Functions and Effective Computability*. MIT Press, Cambridge, MA, USA (1987)
26. Wang, H.: Proving Theorems by Pattern Recognition I. *Communications of the ACM* **3**(4), 220–234 (1960). DOI 10.1145/367177.367224