About the Domino Problem for Subshifts on Groups

Nathalie Aubrun, Sebastián Barbieri, Emmanuel Jeandel

About the Domino problem for subshifts on groups

Nathalie Aubrun[1], Sebastián Barbieri[1], and Emmanuel Jeandel[2]

[1]LIP, ENS de Lyon – 46 allée d'Italie – 69007 Lyon France
[2]LORIA – Campus Scientifique - BP 239 – 54506 Vandoeuvre-Les-Nancy France

February 21, 2024

Abstract

From a classical point of view, the domino problem is the question of the existence of an algorithm which can decide whether a finite set of square tiles with colored edges can tile the plane, subject to the restriction that adjacent tiles share the same color along their adjacent edges. This question has already been settled in the negative by Berger in 1966, however, these tilings can be reinterpreted in dynamical terms using the formalism of subshifts of finite type, and hence the same question can be formulated for arbitrary finitely generated groups. In this chapter we present the state of the art concerning the domino problem in this extended framework. We also discuss different notions of effectiveness in subshifts defined over groups, that is, the ways in which these dynamical objects can be described through Turing machines.
NOTICE: This is the author's version of a work accepted for publication by Springer. Changes resulting from the publishing process, including peer review, editing, corrections, structural formatting and other quality control mechanisms, may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. The definitive version has been published in the book Sequences, Groups, and Number Theory *DOI: 10.1007/978-3-319-69152-7_9*

Contents

1 Introduction **3**

2 Subshifts of finite type on $\mathbb{Z}^2$, Wang tiles and the domino problem **5**
2.1 Definitions . . . 5
2.2 Turing machines and the Halting problem . . . 5
2.3 Reductions . . . 8
2.4 Domino problem with constrained origin . . . 9
2.5 Domino problem . . . 10

3 Subshifts of finite type on finitely generated groups **14**
3.1 Definitions . . . 14
3.1.1 Group presentations and the word problem . . . 14
3.1.2 SFT on finitely generated groups . . . 15
3.2 Domino Problem . . . 17
3.2.1 Definitions . . . 17
3.2.2 Basic properties . . . 18
3.3 Inheritance properties . . . 19
3.4 Classes of groups . . . 21
3.4.1 Virtually free groups . . . 21
3.4.2 Polycyclic groups . . . 21
3.4.3 Baumslag-Solitar groups . . . 22
3.4.4 Groups $G_1 \times G_2$ . . . 25
3.5 Discussion . . . 27
3.5.1 Muller & Schupp theorem . . . 27
3.5.2 Hyperbolic groups . . . 28
3.5.3 Translation-like and quasi-isometric groups . . . 28

4 Towards a definition of effective subshifts on groups **29**
4.1 Link between $\mathbb{Z}$ and $\mathbb{Z}^2$ . . . 29
4.1.1 Projective subdynamics: definition and example . . . 29
4.1.2 Effectively closed subshifts on $\mathbb{Z}^d$ . . . 30
4.1.3 Simulation theorem . . . 32
4.2 Effectiveness on groups . . . 33
4.2.1 Definition and basic properties . . . 33
4.2.2 The case of non recursively presented groups . . . 36
4.2.3 The one-or-less subshift $X_{\leq 1}$ . . . 36
4.3 Two larger notions of effectiveness . . . 37
4.3.1 G-effectiveness . . . 37
4.3.2 Enumeration effectiveness . . . 39
4.3.3 Towards a simulation theorem . . . 41

5 Exercises **42**

1 Introduction

Symbolic dynamics is the study of a particular type of dynamical systems which are called shift spaces or subshifts. These systems can be defined as sets of colorings of a group G by a finite alphabet A which are closed under the product topology and invariant under the shift action induced by the group. These objects were first introduced as a tool to study dynamical systems through discretization in the work of Hadamard [19] and then largely popularized in the highly influential article by Morse and Hedlund [20] where they were studied not only as tools but as inherently interesting objects.

A fundamental property of subshifts is the fact that they can be defined in a purely combinatorial way. Namely, a set of colorings of a group G by a finite alphabet A is a subshift if and only if it can be defined as the set of configurations which avoid a list of forbidden patterns. This motivates the notion of subshift of finite type (SFT), that is, the set of subshifts which can be defined through a finite number of forbidden patterns.

Subshifts of finite type are of high interest from a computational point of view since they can be described by a finite amount of information – a finite set of forbidden patterns that defines the subshift – and thus decidability and algorithmic questions arise naturally. For instance, given a finite set of forbidden patterns F, the simplest question one can formulate is the following: does the subshift X_F defined by F contain at least one configuration? The main goal of this chapter is to present the state of the art concerning that question.

The domino problem of a finitely generated group G (sometimes noted $\mathtt{DP}(G)$ in the sequel) asks essentially the following: is there an algorithm that takes as input a coding of a finite set of forbidden patterns F and outputs `Yes` if the SFT defined by F is non-empty and `No` otherwise? In the particular case where the group G is $\mathbb{Z}$, this problem is decidable: one-dimensional SFTs can be represented by a finite graph [36], and the existence of a configuration in the SFT (i.e. an infinite word) is equivalent to the existence of an infinite path in the graph. The two-dimensional case is much more interesting, since SFTs lack a good graph representation as it exists in 1D, even if some generalizations exist [43, 38].

In the 2D case, the emptiness problem for SFTs is equivalent to the problem of tiling the plane with Wang tiles. A Wang tile is a unit square with a color on each side, that cannot be rotated or reflected. The domino problem $\mathtt{DP}(\mathbb{Z}^2)$ is the algorithmic question of whether a given finite set of Wang tiles can be arranged along a $\mathbb{Z}^2$-lattice in such a way that adjacent tiles have the same color on their adjacent edges. This model is just another way to express local constraints: a set of tilings by Wang tiles can be seen as an SFT – the finite set of tiles stands for the finite alphabet – with constraints on the adjacent tiles. Reciprocally and up to a local recoding of the alphabet, an SFT can be transformed into a set of tilings by Wang tiles.

Originally, the domino problem was formulated on $\mathbb{Z}^2$ by Wang [58] as a toy problem to study a fragment of first order logic (FO). He conjectured that every non-empty SFT on $\mathbb{Z}^2$ admits a periodic configuration, which implies the decidability of domino problem on $\mathbb{Z}^2$. His conjecture was proven wrong by Berger [7] who both exhibited an aperiodic set of 104 Wang tiles and proved the undecidability of the domino problem. This construction, later simplified by Robinson [48], proceeds by reduction from the halting problem of Turing machines.

The domino problem on other structures than $\mathbb{Z}$ and $\mathbb{Z}^2$ has also been successfully investigated. Robinson did not manage to prove undecidability of the problem on the hyperbolic plane, but obtained as a preliminary step the undecidability of the origin constrained version [49] –in this weaker version, noted `OCDP` for Origin constrained domino problem, one asks whether there exists in the SFT a configuration with a given letter at the origin. The undecidability of the unconstrained problem on the hyperbolic plane was proven later by Kari [29], and can also be obtained from the construction of a hierarchical aperiodic tiling on the hyperbolic plane by Goodman-Strauss [18].

For finitely generated groups, the question was formulated as such only very recently. For now no characterization of the groups which have decidable `DP` is known, and the problem seems very difficult to solve. Nevertheless a sufficient condition for decidability of `DP` exists: virtually free groups have decidable domino problem. The fact that they are indeed the only groups where we know the domino problem is decidable motivates the following conjecture: a group has decidable domino problem if and only if it is virtually free. This is further motivated by the following result: if a group is not virtually free, it has a thick end [60] and

then arbitrarily large grids as minors by Halin's theorem (see [15] for a recent proof). It should then be possible to somehow use these grids as computation zones – similarly to what is done in Robinson's tiling [48] – to encode Turing machine computations and use them to obtain the undecidability of DP. But the main problem is that even if we know that such grids exist, we do not know where they appear and even less how to code them using local rules. Recent preprints support the conjecture: to our knowledge, the only other results are that decidability of DP is a quasi-isometry invariant for finitely presented groups [13] – i.e. a geometric property of the group – and that the conjecture holds true for Baumslag-Solitar groups [2], polycyclic groups [23] and groups of the form $G_1 \times G_2$ [24].

To better understand SFTs, we can study them through the prism of projective subdynamics. This operation modifies the group G on which a subshift X is defined into one of its subgroups H: starting from a G-subshift X, the subshift $\pi_H(X)$ is defined as the set of configurations of X restricted to H. For instance one may consider the set of rows that appear in a $\mathbb{Z}^2$-subshift. What can be said about projective subdynamics of SFTs? Addressed this way, this question is unfortunately hard to solve, even for $\mathbb{Z}^2$-SFTs. No complete characterization is known, even if some partial results exist [45]. Nevertheless, if we allow the initial subshift to be sofic –the image of an SFT under a contiguous and shift-commuting map– we get a strong result, known as the simulation theorem. Initially proven by Hochman [21] for sofic $\mathbb{Z}^3$-subshifts, and then generalized to $\mathbb{Z}^2$-subshifts independently in [16] and [3], this result states that the class of projective subdynamics of sofic $\mathbb{Z}^3$ or $\mathbb{Z}^2$-subshifts coincides exactly with the class of effectively closed subshifts, i.e. subshifts that can be defined by a recursively enumerable set of forbidden patterns. This result motivates the study of these objects.

The chapter is organized as follows. Section 2 presents the standard background in dimension 2, and explains where undecidability comes from for the domino problem on $\mathbb{Z}^2$. The heart of the chapter is Section 3, that surveys all existing results about the decidability or undecidability of the domino problem for finitely generated groups. Finally Section 4 is a reflexion about the notion of effectively closed subshift on a finitely generated group. Three different notions of effectiveness are defined, studied and compared.

2 Subshifts of finite type on $\mathbb{Z}^2$, Wang tiles and the domino problem

A Wang tile is a unit square with a color on each side, that cannot be rotated or reflected. In order to tile the plane, Wang tiles can be arranged side by side only if the colors on their adjacent sides match. With this model of tilings in hand, one may wonder whether a given finite set of Wang tiles can tile the entire plane or not. This problem is known as the domino problem, and was originally formulated by Wang [58] as a toy problem to study the $\forall\exists\forall$ fragment of first order logic. He conjectured that every finite set of Wang tiles that can tile the entire plane can also do it in a periodic way, which implies the decidability of domino problem. His conjecture was proven wrong by Berger [8, 7] who both exhibited an aperiodic set of 104 Wang tiles and proved the undecidability of the domino problem. The construction, later simplified by Robinson [48], consists in a tile set that forces all possible tilings to contain a hierarchy of arbitrarily big squares – the plane can be decomposed in squares of order n which are themselves obtained by gluing together the squares of order $n-1$ and so on; aperiodicity comes from the fact that every translation cannot leave all levels of the structure invariant. These squares are then used as computation zones to run one arbitrary Turing machine $\mathcal{M}$; the final set of tiles associated with the Turing machine $\mathcal{M}$ has the property of being non-empty if and only if the machine $\mathcal{M}$ halts on the empty entry. By reduction from the halting problem, we conclude the undecidability of the domino problem.

In this section, we do not present Robinson's construction, but a proof due to Kari [28] that can be generalized to Baumslag-Solitar groups, as described in Section 3.4.3. This alternative proof uses an encoding of rational piecewise affine maps into Wang tiles, and undecidability of the domino problem follows from a reduction to the Mortality problem for piecewise affine maps.

2.1 Definitions

A *Wang tile* is a 4-tuple $t = (t_N, t_W, t_S, t_E) \in C^4$ where C is a finite set. It represents a unit square whose edges are colored according to the tuple interpreting the letters N, S, W, E as north, south, west and east respectively. See Figure 1.

Figure 1: If the set C is to be interpreted as a finite set of colors, a Wang tile defined by a tuple (t_N, t_W, t_S, t_E) of colors can be represented as shown.

A set $\tau \subset C^4$ of Wang tiles is called a *tileset*. We say $x : \mathbb{Z}^2 \to \tau$ is a valid tiling of the plane by τ if and only if for every $(i, j) \in \mathbb{Z}^2$:

$$x(i,j)_N = x(i,j+1)_S \text{ and } x(i,j)_E = x(i+1,j)_W.$$

Said otherwise, a valid tiling is an assignment of tiles from τ to every position of $\mathbb{Z}^2$ such that adjacent Wang tiles share the same color over adjacent edges.

A natural question which arises from this setting is the following: Is there a finite procedure which takes as input a tileset τ and decides whether there exists a valid tiling of the plane? In the next section we introduce the formal concepts needed to precisely state this question.

2.2 Turing machines and the Halting problem

In this section we give some classical definitions, that can be found with more details in [54]. Turing machines were initially introduced by Alan Turing [57] as a mathematical model that would serve for representing computations made by a human being. It is commonly accepted that Turing machines exactly catch what human can compute. This constitutes the *Church-Turing thesis*, which is the hypothesis under which functions computable by Turing machines are exactly functions computable by a human being – with no memory nor time limitation.

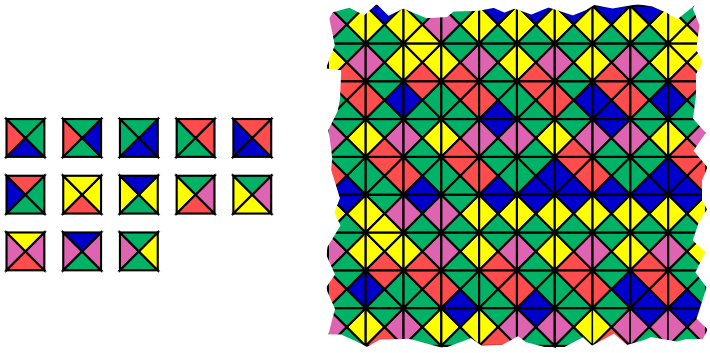


Figure 2: A tileset τ and a partial valid tiling of the plane.

Turing machines are similar to finite automata, except that they can use an unlimited memory with a read/write access. The memory is realized by an infinite tape divided into cells, each cell carrying a symbol chosen among a finite alphabet. At each step of the computation, the head – i.e. the finite automaton – of the Turing machine can read the content of the tape, and depending on the read symbol and its internal state, the head can do some of the following actions: modify the content of the tape, change its internal state and move to a neighbor cell.

A *Turing machine* is a tuple $(Q, \Gamma, \Sigma, \sharp, \delta, q_0, q_a, q_r)$ where Q, Σ are finite sets and

- Q is the set of states,
- Γ is a finite alphabet, the tape alphabet,
- $\Sigma \subset \Gamma$ is the input alphabet,
- $\sharp \in \Gamma \setminus \Sigma$ is the blank symbol,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{-1, 0, 1\}$ is the transition function,
- $q_0 \in Q$ is the initial state,
- q_a and q_r are the accepting and rejecting states, with $q_a \neq q_r$.

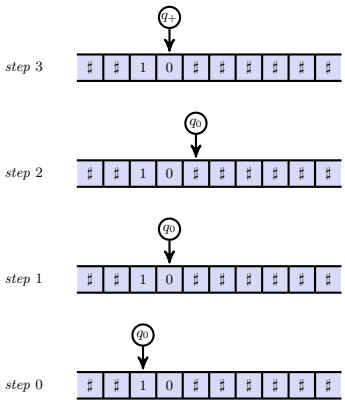
A *configuration of the Turing machine* $\mathcal{M}$ is a tuple (x, i, q) where $x \in \Sigma^{\mathbb{Z}}$ is an infinite tape, $i \in \mathbb{Z}$ is the position of the computation head and q its state. If w is a finite word, we will write (w, i, q) for the configuration where the tape is filled with blank symbols, except on positions $0 \dots |w| - 1$ where the word w is written.

Given a configuration $C = (x, i, q)$, the machine $\mathcal{M}$ computes on C as follows. If $\delta(q, x) = (q', x', \epsilon)$, then the machines goes to configuration $(y, i + \epsilon, q')$ where $y_n = x_n$ for all $n \neq i$ and $y_i = x'$. When the Turing machine $\mathcal{M}$ can go from configuration C to C' in one step, we denote $C \xrightarrow{\mathcal{M}} C'$.

We say that the Turing machine $\mathcal{M}$ accepts (resp. rejects) an input word $w \in \Sigma^*$ if starting from configuration $(w, 0, q_0)$, the machine reaches the accepting state q_a (resp. rejecting state q_r) after a finite number of steps of computation. Given an input word $w \in \Sigma^*$, the machine $\mathcal{M}$ thus has three possible behaviors: it accepts, rejects or runs infinitely (loops). If $\mathcal{M}$ does not run infinitely, it is said to halt on w.

Example 1 *An example of Turing machine and the first steps of a computation starting with configuration* $(\ldots\sharp 10\sharp\ldots, 0, q_0)$.

$\delta(q,x)$		*Symbol* x		
		0	1	$\sharp$
State q	q_0	$(q_0, 0, 1)$	$(q_0, 1, 1)$	$(q_+, \sharp, -1)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, -1)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



This Turing machine has the following behavior. If there is a finite number of symbols 0*'s and* 1*'s around the computation head on the initial tape, then the machine adds* 1 *to the number coded in binary on the tape, and then halts. Otherwise, the machine never halts.*

In our definition, the tape is fixed and the computation head moves, but Turing machines can equivalently be defined with a fixed computation head and a moving tape. This variant is called *moving tape Turing machine* [32], and configurations in this model are of the form (x, q) where $x \in \Sigma^{\mathbb{Z}}$ is an infinite tape and $q \in Q$ is the current state.

By definition, there are at most countably many Turing machines, and each given Turing machines can be encoded in a finite word, which is denoted $\langle \mathcal{M} \rangle$. More generally, we denote $\langle a, b \rangle$ the word that encodes the pair of objects (a, b) – objects can be words, Turing machines, or everything that possesses a finite description.

A language L is *decidable* if there exists a Turing machine $\mathcal{M}$ such that $\mathcal{M}$ accepts a word w if $w \in L$ and rejects w if $w \notin L$. A language L is *recursively enumerable* if there exists a Turing machine $\mathcal{M}$ such that $\mathcal{M}$ accepts a word w if and only if $w \in L$ – it can reject or loop otherwise.

A *decision problem* is a problem that takes an input that can be encoded into a finite word, and has a `Yes/No` answer. A decision problem is decidable if the language of encodings of its inputs with positive answer is decidable, undecidable otherwise.

A natural decision problem about Wang tiling is the so-called domino problem.

Definition 1 *The* domino problem *is the decision problem that takes as input a finite tileset* τ*, and outputs* `Yes` *if and only if there exists a valid tiling of the plane by* τ*.*

Wang originally conjectured that if a set of Wang tiles can tile the plane, then they can always be arranged to do so periodically. Here by periodic tiling we mean that the tiling can be constructed by repeating a rectangular pattern, where occurrences of this pattern are arranged on a sublattice of $\mathbb{Z}^2$. If this conjecture were true, then we could decide the domino problem by running in parallel the two following semi-algorithms – procedures that do not necessarily halt. The first semi-algorithm searches for a periodic rectangular pattern in the sense given above, by enumerating valid rectangular patterns by increasing size, and checking if the sequences of colors that label the North and South edges (resp. West and East edges) match up to a cyclic permutation. The second semi-algorithm tries to tile bigger and bigger squares by a brute-force strategy. The first semi-algorithm halts if and only if there exists a periodic pattern; the second halts if and only if the set of tiles cannot tile the plane. Thus Wang's conjecture implies the decidability of the domino problem. In other words, the undecidability of the domino problem implies the existence of a set of Wang tiles that tiles the plane, but never in a periodic way – such sets of tiles are called aperiodic sets of tiles. Wang's conjecture was disproven by Berger [8], who proved the undecidability of the domino problem. It is noteworthy that his proof relies on the construction of an aperiodic set of tiles.

Theorem 2 (Berger, [7, 8]) *The domino problem is undecidable.*

A detailed proof of the undecidability of the domino problem will be given in Section 2.5.

Definition 2 *The* Halting problem *is the decision problem that takes as input a Turing machine $\mathcal{M}$ and an input word w, and outputs* `Yes` *if and only if the machine $\mathcal{M}$ reaches a final state during its computation on w.*

Theorem 3 (Turing, [57]) *The Halting problem is undecidable.*

Proof: Suppose that the following language

$$HALT = \{\langle \mathcal{M}, w\rangle \mid \mathcal{M} \text{ halts on } w\}$$

is decidable. Then there exists a Turing machine $\mathcal{H}$ with the following behavior: $\mathcal{H}$ accepts the entry $\langle \mathcal{M}, w\rangle$ if $\mathcal{M}$ halts on w, and rejects $\langle \mathcal{M}, w\rangle$ if $\mathcal{M}$ loops on w. We construct a Turing machine $\mathcal{N}$ that uses $\mathcal{H}$ as a subroutine. More precisely, on the input $\langle \mathcal{M}\rangle$, the machine $\mathcal{N}$ runs $\mathcal{H}$ on the input $\langle \mathcal{M}, \langle \mathcal{M}\rangle\rangle$, accepts if $\mathcal{H}$ rejects and loops if $\mathcal{H}$ accepts. Running $\mathcal{N}$ on its own coding $\langle \mathcal{N}\rangle$ leads to a contradiction, since the machine $\mathcal{N}$ should both accept and loop ! Thus the machine $\mathcal{H}$ cannot exist. ■

In what precedes, Turing machines are seen as a device that can accept, reject or loop on a given input. Another way to use this computational model is to consider the finite word written on the tape when a final state is reached as the output of the machine. A function $f : D \subseteq \{0,1\}^* \to \{0,1\}^*$ is *computable* if there exists a Turing machine $\mathcal{M}$ with the following behavior: if $w \in D$, then the machine accepts on entry w, and the tape contains $f(w)$ when the final state is reached.

2.3 Reductions

In what follows, $\overline{L}$ denotes the complement of the language L.

Reductions aim at comparing the computational difficulty of decision problems. In this section we present two of them which are meaningful for this chapter, Turing reduction and many-one reduction. In Section 4.3.2 a third one will be introduced, the enumeration reduction.

An *oracle Turing machine* is a couple $(\mathcal{M}, L)$, where $\mathcal{M}$ is a classical Turing machine with an additional state called the oracle state, and $L \subset \Gamma^*$ is a language called the oracle, that can be queried in a single step of computation. When that machine $\mathcal{M}$ enters its oracle state, it can asks whether the word written on its tape belongs to L or not, and then evolves according to the answer. The oracle does not need to be a recursive or recursively enumerable language, so that the addition of an oracle may increase the computational power of the model. We will not explain in details how the oracle can be formalized as an extension of a classical Turing machine, but the reader can find details in [54].

Definition 3 *The language L is* Turing reducible *to L', denoted $L \leq_T L'$, if there exists a Turing machine with oracle L' that computes L. We note $L \equiv_T L'$ if $L \leq_T L'$ and $L' \leq_T L$.*

Example 4 *One has that $HALT \leq_T \overline{HALT}$. Indeed, consider the Turing machine with oracle $\overline{HALT}$ that immediately requests the oracle on its input word, and then returns the negation of the oracle result. This machine accepts an input word $\langle \mathcal{M}, w\rangle$ if $\langle \mathcal{M}, w\rangle \notin \overline{HALT}$ and rejects otherwise. So it is a Turing machine with oracle $\overline{HALT}$ that computes $HALT$.*

Definition 4 *The language L is* many-one reducible *(also called* mapping reducible *in [54]) to L', denoted $L \leq_m L'$, if there exists a computable function f such that $x \in L$ iff $f(x) \in L'$ for every x. We note $L \equiv_m L'$ if $L \leq_m L'$ and $L' \leq_m L$.*

Example 5 *One has that $HALT \not\leq_m \overline{HALT}$. Indeed, suppose $HALT \leq_m \overline{HALT}$, that is to say there exists a computable function f such that $x \in HALT$ iff $f(x) \in \overline{HALT}$ for every x. We construct the Turing machine $\mathcal{M}_m$ as follows. On an input $\langle \mathcal{M}, w\rangle$, it first computes the word $f(\langle \mathcal{M}, w\rangle) = \langle \mathcal{M}', w'\rangle$. Then the machines simulates in parellel –one step for each simulation– the machine $\mathcal{M}$ on w and the machine $\mathcal{M}'$ on w'. If $\mathcal{M}$ halts on w, then the machine $\mathcal{M}_m$ accepts. If $\mathcal{M}'$ halts on w', the machine $\mathcal{M}_m$ rejects. One can check that $\mathcal{M}_m$ computes $HALT$, raising a contradiction.*

One can show that many-one reducibility is stronger than Turing reducibility (see Exercise 1).

Definition 5 *The* Blank tape Halting problem *is the decision problem that takes as input a Turing machine $\mathcal{M}$, and outputs* `Yes` *if and only if the machine $\mathcal{M}$ reaches a final state during its computation initiated on the empty tape*

$$HALT_b = \{\langle \mathcal{M} \rangle \mid \mathcal{M} \text{ halts on the empty input}\}.$$

As an example of Turing reduction, we show the following.

Proposition 6 (Folklore) *The Blank tape Halting problem is undecidable.*

Proof: We prove that $HALT_b \leq_T HALT$, which is enough to prove that the Blank tape Halting problem is undecidable. The fundamental ingredient in this proof is that the encoding ($a \mapsto \langle a \rangle$ or $(a, b) \mapsto \langle a, b \rangle$) and decoding ($\langle a \rangle \mapsto a$ or $\langle a, b \rangle \mapsto (a, b)$) functions of finite objects are computable.

Let $\mathcal{M}_b$ be the Turing machine with oracle $HALT$ with the following behavior. On a given input word m, it first decodes m as a $\langle \mathcal{M} \rangle$, and then encodes $\langle \mathcal{M}, \varepsilon \rangle$ as a new word w that is now written on the tape (ε denotes the empty word). The machine now changes its state to enter the oracle state: if the word w written on the tape belongs to $HALT$ then the machines accepts, otherwise it rejects. Thus the language $HALT_b$ is computed by the machine $\mathcal{M}_b$ with oracle $HALT$. ■

2.4 Domino problem with constrained origin

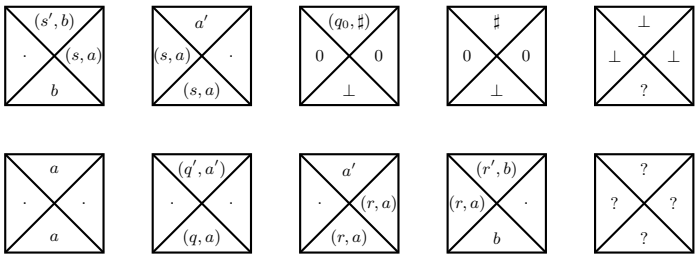
As explained above, the behavior of a Turing machine only depends on local information (the state of the head and the content of the tape). Consequently, this is relatively easy to encode it inside a finite set of Wang tiles. We give a concrete encoding of the behavior of a given Turing machine $\mathcal{M}$ inside a finite tileset $\tau_{\mathcal{M}}$.

Definition 6 *The* Origin Constrained domino problem *is the decision problem that takes as input a finite tileset τ and a tile $t \in \tau$, and outputs* `Yes` *if and only if there exists a valid tiling of the plane by τ with the tile t at the origin.*

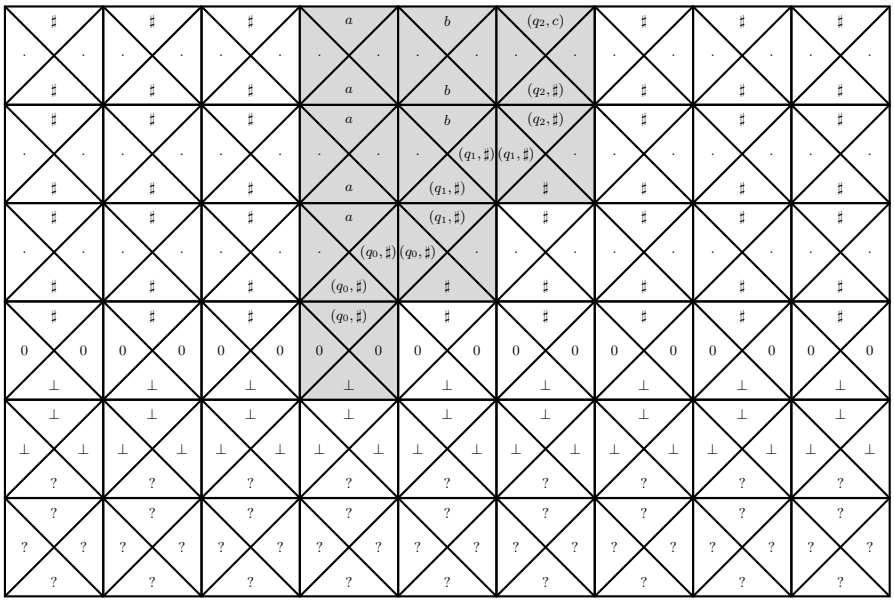
Remark 7 *Suppose the Origin Constrained domino problem is decidable, and fix a finite tileset τ. Then it suffices to run the corresponding algorithm successively on the inputs (τ, t) for every tile $t \in \tau$ to get the decidability of the domino problem.*

Theorem 8 (Kahr, Moore & Wang [27], Büchi [10]) *The Origin Constrained domino problem is undecidable.*

Proof: Let $\mathcal{M}$ be a Turing machine. Consider the following tileset $\tau_{\mathcal{M}}$, where tiles are defined for every $a \in \Sigma$, for every $(q, a) \in Q \times \Sigma$ such that $\delta(q, a) = (q', a', .)$, for every $(r, a) \in Q \times \Sigma$ such that $\delta(r, a) = (r', a', 1)$ and for every $(s, a) \in Q \times \Sigma$ such that $\delta(s, a) = (s', a', -1)$:



Impose that the tile $t_0 =$  appears at the origin. Then there is exactly one way to fill in the first row and the half-plane below it. Morevoer, tiles located strictly to the left of the origin are also uniquely determined. Thus only the filling of upper-right quadrant depends on the Turing machine $\mathcal{M}$.



Suppose that the tiling presented above has been extended to a tiling until the ith row. Then on the top edge of row i, one can read the configuration $C_i = (w_0 \dots w_n \dots, j, q)$, the ith configuration of $\mathcal{M}$ on ε. So the tiling can be extended to row $i+1$ if and only if there exists a configuration $C_{i+1} = Next(C_i)$. Thus the computation of $\mathcal{M}$ on ε is infinite if and only if there exists a tiling by $\tau_{\mathcal{M}}$ with tile t_0 at the origin. Since the Blank tape Halting problem is undecidable, we conclude the Origin constrained domino problem is undecidable. ■

2.5 Domino problem

The undecidability of the domino problem was originally proven by Berger [8]. We present here an alternative proof, given by Kari [28], that has one main advantage for the purpose of this chapter: the construction can be adapted to other groups than $\mathbb{Z}^2$ (see Section 3.4.3).

Definition 7 *The* Mortality problem of Turing machines *is the decision problem that takes as input a deterministic Turing machine $\mathcal{M}$ with an halting state, and outputs* `Yes` *if and only if there exists a non-halting configuration – configuration that never evolves into the halting state.*

It is important to note that in this problem, the machine does not start from an initial configuration: the starting state and the starting tape are arbitrary. It is interesting to know that, while the first proof of the undecidability of the domino Problem comes from Berger, a student of Wang, the main ingredient for this new proof is from another student of Wang. Technical details can be found in [22].

Theorem 9 (Hooper, [22]) *The Mortality problem of Turing machines is undecidable.*

The proof proceeds by several reductions: the immortality problem of Turing machines reduces to the immortality problem of 4-counters machines, that itself reduces to the halting problem of 2-counters machines, that finally reduces to the halting problem for Turing machines, which is undecidable by Theorem 3. Note that the undecidability of the Mortality problem for reversible Turing machines, a stronger result, was proven in [30] with a much simpler proof.

Given a system of rational affine transformations of the plane $f_1, f_2, \dots, f_n$ associated with disjoint unit squares $U_1, U_2, \dots, U_n$ with integer corners, we define a partial function $f : \mathbb{R}^2 \to \mathbb{R}^2$ with domain $U = \cup_{i=1}^{n} U_i$ given 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* if for every $n \in \mathbb{N}$, the point $f^n(\overrightarrow{x})$ lies inside the domain U.

Definition 8 *The* Mortality problem of piecewise affine maps *is the decision problem that takes as input a system of rational affine transformations of the plane $f_1, f_2, \ldots, f_n$ associated with disjoint unit squares $U_1, U_2, \ldots, U_n$ with integer corners, and outputs* `Yes` *if and only if the system has an immortal starting point.*

Theorem 10 ([28]) *The Mortality problem of piecewise affine maps is undecidable.*

Proof: Given a Turing machine $\mathcal{M}$, we construct a system of piecewise affine maps that has an immortal starting point if and only if $\mathcal{M}$ has an immortal configuration. The construction presented here is the one from [28], and refers to [31, 9]. We assume that the machine $\mathcal{M}$ is a moving tape machine (see Section 2.2), and that its states and alphabet are $A = \{0, 1, \ldots, a-1\}$ and $Q = \{0, 1, \ldots, b-1\}$. The current configuration (x, q) of the machine will be coded by the two real numbers

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

and

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

where M is an integer such that $M > a$ and $M > b-1$. The integer $\lfloor r \rfloor = Mq + x_0$ is enough to determine the next configuration, and a transition of the Turing machine corresponds to an affine map with matrix

$$\begin{pmatrix} M & 0 \\ 0 & \frac{1}{M} \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \text{ or } \begin{pmatrix} \frac{1}{M} & 0 \\ 0 & M \end{pmatrix}$$

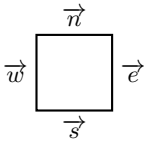
depending on the tape movement. The translation constant of the affine map is adjusted to code the change of state and the change of the symbol on the tape. For instance, the transition $\delta(q, a) = (q', a', 1)$ is coded by the affine transformation

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

The domain of this affine map is the unit square with integer coordinates $[0, 1] \times [Mq, Mq+1]$. With this procedure we transform a Turing machine $\mathcal{M}$ into a finite set of rational affine transformations $f_1, \ldots, f_n$ and disjoint unit squares with integer coordinates $U_1, \ldots, U_n$. One can check that immortality is preserved under this transformation: the Turing machine $\mathcal{M}$ has an immoral configuration if and only if the system of affine maps $f_1, \ldots, f_n$ has an immortal point. From Theorem 9 we conclude the the immortality problem of piecewise affine maps is undecidable. ■

Theorem 11 *The domino problem is undecidable.*

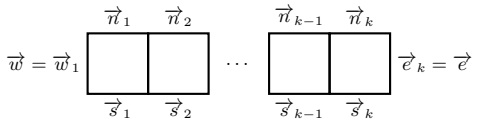
Proof: We present the proof due to Kari [28], that proceeds by reduction from the Mortality problem of piecewise affine maps. Consider $f : \mathbb{R}^2 \to \mathbb{R}^2$ a rational affine map. We construct a finite set of Wang tiles τ_f whose colors are chosen in $\mathbb{R}^2$. We first give an idea of how the tileset is made, and will explain further how colors are chosen to get only a finite number of tiles. The tile



is said to compute the affine function f if

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

In other terms, $\vec{n}$ is the input on the top edge, and the output $\vec{s}$ is computed on the bottom edge. The computation is not exact: a carry $\vec{w}$ from the left edge is added to $f(\vec{n})$ and a carry $\vec{e}$ from the right edge is added to $\vec{s}$. Suppose now that a finite portion of a row is tiled with tiles that compute the function f as pictured below.



In the case where f is affine and since $e_i = w_{i+1}$ for $i = 1 \dots k-1$ by matching rules, it follows that

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

The carries will eventually vanish as the size of the finite portion tends to infinity, so that roughly speaking the average of the bottom labels will be the image by f of the average of the top labels.

Let f_i be the rational affine map with domain $U_i = [n, n+1] \times [m, m+1]$, given by

$$f_i(\overrightarrow{x}) = M\overrightarrow{x} + \overrightarrow{b}.$$

To describe the finite set of tiles that encodes f_i, we need some additional definitions. For $\overrightarrow{x} \in \mathbb{R}^2$ and $k \in \mathbb{Z}$, denote $A_k(\overrightarrow{x}) = \lfloor k\overrightarrow{x} \rfloor$, where $\lfloor (x, y) \rfloor = (\lfloor x \rfloor, \lfloor y \rfloor)$. Denote also

$$B_k(\overrightarrow{x}) = A_k(\overrightarrow{x}) - A_{k-1}(\overrightarrow{x}) = \lfloor k\overrightarrow{x} \rfloor - \lfloor (k-1)\overrightarrow{x} \rfloor.$$

If $\overrightarrow{x}$ is in the domain $U_i = [n, n+1] \times [m, m+1]$, one can check that $B_k(\overrightarrow{x}) \in \{(n, m), (n, m+1), (n+1, m), (n+1, m+1)\}$ for every $k \in \mathbb{Z}$. In other words, $(B_k(\overrightarrow{x}))_{k \in \mathbb{Z}}$ is a sequence of elements chosen in $\{(n, m), (n, m+1), (n+1, m), (n+1, m+1)\}$.

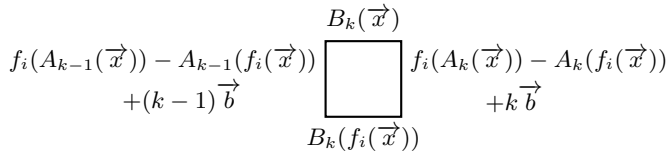
We say that a bi-infinite sequence $(x_k)_{k \in \mathbb{Z}}$ of i's and $(i+1)$'s represents a real number $x \in [i, i+1]$ if there exists a sequence of intervals $I_1 \subset I_2 \subset \cdots \subset \mathbb{Z}$ of increasing lengths $n_1 < n_2 < \dots$ such that

$$\lim_{k \to \infty} \frac{\sum_{j \in I_k} x_j}{n_k} = x,$$

that is to say there is an infinite sequence of intervals of increasing lengths whose averages converge to x. Note that if $(x_k)_{k \in \mathbb{Z}}$ is a representation of x, all the shifted sequences $(x_{\ell+k})_{k \in \mathbb{Z}}$ for every $\ell \in \mathbb{Z}$ are also representations of x. Note also that a sequence $(x_k)_{k \in \mathbb{Z}}$ can represent several distinct real numbers, since different interval sequences may converge to different points, and that by a compactness argument, every sequence $(x_k)_{k \in \mathbb{Z}}$ does represent at least one real number x.

Clearly the bi-infinite sequence $(B_k(\overrightarrow{x}))_{k \in \mathbb{Z}}$ is a representation of $\overrightarrow{x}$ in the sense defined above. It is called a balanced representation of $\overrightarrow{x}$.

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



for every $k \in \mathbb{Z}$ and $\overrightarrow{x} \in U_i$. One can check that these tiles compute the function f_i.

$$\begin{aligned}
\delta &= f_i(\overrightarrow{n}) + \overrightarrow{w} - \overrightarrow{s} - \overrightarrow{e} \\
&= f_i\left(B_k(\overrightarrow{x})\right) + f_i\left(A_{k-1}(\overrightarrow{x})\right) - A_{k-1}\left(f_i(\overrightarrow{x})\right) + (k-1)\overrightarrow{b} - B_k\left(f_i(\overrightarrow{x})\right) \\
&\quad - f_i\left(A_k(\overrightarrow{x})\right) + A_k\left(f_i(\overrightarrow{x})\right) - k\overrightarrow{b} \\
&= M\lfloor k\overrightarrow{x} \rfloor - M\lfloor (k-1)\overrightarrow{x} \rfloor + \overrightarrow{b} + M\lfloor (k-1)\overrightarrow{x} \rfloor + \overrightarrow{b} - \lfloor (k-1)f_i(\overrightarrow{x}) \rfloor \\
&\quad + (k-1)\overrightarrow{b} - \lfloor kf_i(\overrightarrow{x}) \rfloor + \lfloor (k-1)f_i(\overrightarrow{x}) \rfloor - M\lfloor k\overrightarrow{x} \rfloor - \overrightarrow{b} + \lfloor kf_i(\overrightarrow{x}) \rfloor - k\overrightarrow{b} \\
&= 0.
\end{aligned}$$

Denote by A_i the finite alphabet used to color the edges of tiles in τ_{f_i}. Since the domain U_i is bounded, there are only finitely many possible values for the top and bottom colors in the tileset. The case of left and right colors is a little bit more subtle. Denote

$$c_k(\overrightarrow{x}) = f_i(A_k(\overrightarrow{x})) - A_k(f_i(\overrightarrow{x})) + k\overrightarrow{b}$$

the color on the right edge of the tile of the figure above, so that the left color is $c_{k-1}(\overrightarrow{x})$. By using the fact that

$$\overrightarrow{x} - \overrightarrow{1} \leq \lfloor \overrightarrow{x} \rfloor < \overrightarrow{x}$$

for every $\overrightarrow{x} \in \mathbb{R}^2$, where $\overrightarrow{1}$ denotes the vector $(1,1)$, we get that

$$M(k\overrightarrow{x} - \overrightarrow{1}) + \overrightarrow{b} - k(M\overrightarrow{x} + \overrightarrow{b}) + k\overrightarrow{b} \leq c_k(\overrightarrow{x}) \leq Mk\overrightarrow{x} + \overrightarrow{b} - k(M\overrightarrow{x} + \overrightarrow{b}) + \overrightarrow{1} + k\overrightarrow{b}$$
$$-M \cdot \overrightarrow{1} + \overrightarrow{b} \leq c_k(\overrightarrow{x}) \leq \overrightarrow{b} + \overrightarrow{1}.$$

Since $\overrightarrow{b}$ is a rational vector and M has rational coefficients, by taking q the lcm of the denominators of all the rational numbers appearing in $\overrightarrow{b}$ and M, we get the existence of $\overrightarrow{p_1}, \overrightarrow{p_2} \in \mathbb{Z}^2$ such that

$$\frac{\overrightarrow{p_1}}{q} \leq c_k(\overrightarrow{x}) \leq \frac{\overrightarrow{p_2}}{q},$$

where $\overrightarrow{p_1}$ is chosen maximal and $\overrightarrow{p_2}$ minimal. And even better than that, it happens that all values $c_k(\overrightarrow{x})$ are in the finite set

$$\left\{ \frac{\overrightarrow{p_1}}{q}, \frac{\overrightarrow{p_1} + (0,1)}{q}, \frac{\overrightarrow{p_1} + (1,0)}{q}, \frac{\overrightarrow{p_1} + \overrightarrow{1}}{q}, \ldots, \frac{\overrightarrow{p_2}}{q} \right\} \subset \mathbb{Q}$$

for every $k \in \mathbb{Z}$ and every $\overrightarrow{x} \in U_i$. Indeed, a careful observation of all rational numbers that appear inside the expression of $c_k(\overrightarrow{x})$ shows that it can be written as $\frac{\overrightarrow{p}}{q}$, and the fact that $\overrightarrow{p_1} \leq \overrightarrow{p} \leq \overrightarrow{p_2}$ directly follows from the definition of $\overrightarrow{p_1}$ and $\overrightarrow{p_2}$. So the tileset τ_{f_i} corresponding to f_i is finite. By definition of τ_{f_i}, for a given $\overrightarrow{x} \in U_i$, one can tile a row with τ_{f_i} such that the balanced representations of $\overrightarrow{x}$ and $f_i(\overrightarrow{x})$ appear on the top and bottom labels respectively.

Suppose now that we have a system of rational affine maps $f_1, f_2, \ldots, f_n$ associated with unit squares $U_1, U_2, \ldots, U_n$ with integer corners. From each function f_i we construct a finite set of tiles τ_{f_i} that computes f_i as explained above, whose top colors $\overrightarrow{n}$ are in U_i and bottom colors $\overrightarrow{s}$ in $f_i(U_i)$. We use an additional marking on the tiles –for instance by adding the color $i \in \{1, \ldots, n\}$ to every color from A_i– so that a row can be tiled only with tiles constructed from the same f_i. We get a final finite tileset $\tau_f \subset \bigcup_{i=1}^{n} (A_i \times \{i\})^4$.It remains to prove that the tileset τ_f admits a tiling of the plane if and only if the system $f_1, f_2, \ldots, f_n$ has an immortal point.

Suppose that the system $f_1, f_2, \ldots, f_n$ has an immortal point x in one of the U_i. We construct the tiling $t \in \tau_f^{\mathbb{Z}^2}$ by assigning to every position $(k,j) \in \mathbb{Z}^2$ the tile

$$\begin{array}{ccc} & B_k\left(f^j(\overrightarrow{x})\right) & \\ f\left(A_{k-1}\left(f^j(\overrightarrow{x})\right)\right) - A_{k-1}(f^{j+1}(\overrightarrow{x})) & \square & f\left(A_k\left(f^j(\overrightarrow{x})\right)\right) - A_k(f^{j+1}(\overrightarrow{x})) \\ +(k-1)\overrightarrow{b} & & +k\overrightarrow{b} \\ & B_k\left(f^{j+1}(\overrightarrow{x})\right) & \end{array}$$

which gives a valid tiling in X_{τ_f}. Reciprocally, suppose that τ_f admits a valid tiling of the plane $t \in \tau_f^{\mathbb{Z}^2}$. There is no reason that would force the sequences of top labels that appear on a given row to be the balanced representation of a number $x \in U$. Nevertheless, we can proceed by extraction to prove the existence of an immortal point for f. Consider the intervals $I_k = \{-k, \ldots, k\}$ for all $k \in_{\mathbb{N}}$. Define vectors $(\overrightarrow{x_k})_{k \in \mathbb{N}}$ as follows:

$$\overrightarrow{x_k} = \frac{\sum_{i=-k}^{k} \overrightarrow{n}(t_{i,0})}{2k+1},$$

in other words we look at the mean of increasing sums of top labels of the first row in the tiling t. Be definition of the tileset τ_f, we immediately get that there exists some $1 \leq i \leq n$ such that $\overrightarrow{x_k} \in U_i$ for every $k \in \mathbb{N}$. By compactness of U_i, we extract a sequence $\left(\overrightarrow{x_{\phi(k)}}\right)$ that converges to $\overrightarrow{x} \in U_i$. By continuity of each f_i, we can check that $\overrightarrow{x}$ is an immortal point for f. ■

3 Subshifts of finite type on finitely generated groups

3.1 Definitions

3.1.1 Group presentations and the word problem

Let G be a group. For words $u, v \in G^*$ we write $u =_G v$ if after applying the group operation on each pair of contiguous symbols the same element of G is obtained on both sides.

Definition 9 *Let G be a group and $F \subset G$. The group generated by F is the set*

$$\langle F \rangle := \{g \in G \mid \exists u \in (F \cup F^{-1})^* \text{ such that } u =_G g\}.$$

It is clear that $\langle F \rangle$ is the smallest subgroup of G that contains F.

Definition 10 *We say a group G is* finitely generated *if there exists a finite subset $\mathsf{S} \subset G$ such that $G = \langle \mathsf{S} \rangle$. Such a set S is called a set of* generators *for G. The rank of G is defined as the smallest cardinality of a set of generators for G.*

Example 12 *The group of complex numbers of the form $e^{2i\pi n\alpha}$ for $n \in mathbbZ$ with multiplication as the operation is finitely generated with rank 1. Indeed, it is generated by $e^{2i\pi\alpha}$. Note that this group is infinite if and only if $\alpha \notin \mathbb{Q}$.*

Example 13 *The group $(\mathbb{Q}, +)$ of rational numbers with addition has infinite rank. Indeed, for any set of finite rational numbers $p_1/q_1, \ldots, p_n, q_n$, the denominator of any element of $\langle p_1/q_1, \ldots, p_n, q_n \rangle$ is bounded by $\prod_{i=1}^{n} q_i$. Therefore it cannot generate $\mathbb{Q}$.*

By definition of $\langle \mathsf{S} \rangle$, each element of a finitely generated group can be seen as a word in $(\mathsf{S} \cup \mathsf{S}^{-1})^*$. From now on, we will use the convention that every set of generators contains its inverses to avoid writing $\mathsf{S} \cup \mathsf{S}^{-1}$.

Definition 11 *Let G be a group and $\mathsf{S} \subset G$. The right* Cayley graph *of G with respect to S is the colored directed graph $\Gamma(G, \mathsf{S})$ whose vertex set is G and its set of arcs is given by $E = \bigcup_{s \in \mathsf{S}} E_s$ where E_s is the set of arcs colored by $s \in \mathsf{S}$ defined by $E_s := \{(g, gs) \mid g \in G\}$.*

If S generates G then $\Gamma(G, \mathsf{S})$ is connected. For $g \in G$ we denote $|g|_{\mathsf{S}}$ the length of the shortest path from 1_G to g in $\Gamma(G, \mathsf{S})$. This induces a distance $d_{\mathsf{S}}(g, h) := |g^{-1}h|$. We denote the closed ball centered in $g \in G$ of radius r by $B_{\mathsf{S}}(g, r) = \{h \in G \mid d_{\mathsf{S}}(g, h) \leq r\}$.

Example 14 *Consider the group $\mathbb{Z}^2$ endowed with coordinate-wise sum as the operation. Let $\mathsf{S} = \{(0, 1), (1, 0), (0, -1), (-1, 0)\}$ be the canonical set of generators. Then $\Gamma(\mathbb{Z}^2, \mathsf{S})$ is the bi-infinite grid and $|(n_1, n_2)|_{\mathsf{S}} = |n_1| + |n_2|$ is the taxicab norm.*

Definition 12 *Let S be a set and consider a copy $\mathsf{S}^{-1} = \{s^{-1} \mid s \in \mathsf{S}\}$. We say a word in $(\mathsf{S} \cup \mathsf{S}^{-1})^*$ is* reduced *if it does not contain ss^{-1} or $s^{-1}s$ as subwords. Every word in can be reduced to an unique minimal word by successively eliminating every apparition of ss^{-1} or $s^{-1}s$.*

Definition 13 *The* free group *over S is defined as the group F_{S} of all reduced words in $(\mathsf{S} \cup \mathsf{S}^{-1})^*$ endowed with word concatenation followed by reduction as the operation.*

A more combinatorial way to look at groups is using presentations. A *group presentation* is a pair (S, R) where S is a set and $\mathsf{R} \subset (\mathsf{S} \cup \mathsf{S}^{-1})^*$ is a set of words. Elements of S are called *generators* and words of R are called *relators*.

Definition 14 *Let G be a group. We say (S, R) is a presentation of G if G is isomorphic to $\langle \mathsf{S} | \mathsf{R} \rangle$ where*

$$\langle \mathsf{S} | \mathsf{R} \rangle = F_{\mathsf{S}} / N_{\mathsf{R}}.$$

Here F_{S} is the free group over S and N_{R} is the conjugate closure of R, that is, $N_{\mathsf{R}} = \langle \{grg^{-1} \mid g \in F_{\mathsf{S}} \text{ and } r \in \mathsf{R}\} \rangle$.

In other words, $\langle \mathsf{S} | \mathsf{R} \rangle$ is the largest quotient of the free group over S such that every word in R is identified to the empty word.

Example 15 *We have that $\mathbb{Z}^2 \cong \langle a, b \mid aba^{-1}b^{-1} \rangle$.*

Definition 15 *We say a group G is* recursively presented *if there exists a presentation (S,R) such that $G \cong \langle \mathsf{S}|\mathsf{R}\rangle$, S is recursive and R is a recursively enumerable language. If there exists a presentation for G for which both S and R are finite we say G is* finitely presented.

Definition 16 *The* word problem of a group G *with respect to a set of generators S is the language $\mathtt{WP}(G,\mathsf{S}) = \{u \in \mathsf{S}^* \mid u =_G 1_G\}$.*

Proposition 16 *Let $\mathsf{S}_1, \mathsf{S}_2$ be two finite sets of generators for G. Then $\mathtt{WP}(G,\mathsf{S}_1)$ is many-one equivalent to $\mathtt{WP}(G,\mathsf{S}_2)$.*

Proof: As $\langle \mathsf{S}_2\rangle = G$ we have that each $s \in \mathsf{S}_1$ can be written as $u(s) \in \mathsf{S}_2^*$ such that $s =_G u(s)$. As S_1 is finite, the function which sends a word $s_0 \cdots s_k \in \mathsf{S}_1^*$ to $u(s_0)\cdots u(s_k) \in \mathsf{S}_2^*$ is total computable and $s_0 \cdots s_k = 1_G \iff u(s_0)\cdots u(s_k) = 1_G$.

In view of Proposition 16 in terms of computability we can unambiguously speak about the *word problem* of a group G and denote it as $\mathtt{WP}(G)$.

Proposition 17 *A finitely generated group G is recursively presented if and only if $\mathtt{WP}(G)$ is recursively enumerable.*

Proof: If $\mathtt{WP}(G,\mathsf{S})$ is recursively enumerable one can choose $(\mathsf{S},\mathtt{WP}(G,\mathsf{S}))$ as a presentation for G. Conversely, as G is recursively presented then $G \cong F_{\mathsf{S}}/N_{\mathsf{R}}$ for some recursively enumerable $\mathsf{R} \subset \mathsf{S}^*$. Given $u \in F_{\mathsf{S}}$ we have $u =_G 1_G \iff u \in N_{\mathsf{R}}$, therefore it suffices to be able to recognize this set. An algorithm which does this is the following: Iteratively for each $n \in \mathbb{N}$ run for n steps the algorithm recognizing R on all words on S^* of length at most n. Let A_n be the list of accepted words so far. Build $B_n = \{w\ell w^{-1} \mid |w| < n, \ell \in B_n\}$ and $C_n = \{u \in B_n^* \mid |u| \leq n\}$. The set C_n approximates the conjugate closure of R. It is easy to see that every possible word in N_{R} appears in C_n for large enough n.

3.1.2 SFT on finitely generated groups

Most of the definitions are analogous to the one-dimensional case. Let A be a finite alphabet. The set A^G endowed with the left group action $S : G \times A^G \to A^G$ given by $S^g(x)_h = x_{g^{-1}h}$ is a *full shift*. The elements $a \in A$ and $x \in A^G$ are called *symbols and* configurations respectively. With the product of the discrete topology on A the set of configurations A^G is a compact metric space that has the *cylinders* $[a]_g = \{x \in A^G | x_g = a\}$ *as a subbasis. A* support is a finite subset $F \subset G$. Given a support F, a *pattern with support F* is an element $p \in A^F$ and we write $\mathrm{supp}(p) = F$. We also denote the cylinder generated by p in position g as $[p]_g = \bigcap_{h\in F}[p_h]_{gh}$, and $[p] = [p]_{1_G}$.

Definition 17 *A* subshift *is a subset $X \subset A^G$ which is closed and shift invariant, that is, $S(X) \subset X$. Equivalently a subshift is the set of configurations $X_{\mathcal{F}}$ defined by a set of forbidden patterns $\mathcal{F}$ as follows:*

$$X_{\mathcal{F}} = A^G \setminus \bigcup_{p\in\mathcal{F}, g\in G} [p]_g.$$

Definition 18 *The* language *of a subshift $L(X)$ is the set of patterns p that appear in a configuration of X, that is, $[p] \cap X \neq \emptyset$. In particular $L(A^G)$ is the set of all patterns.*

Let $X \subset A^G$ and $Y \subset B^G$ be subshifts. A continuous map $\sigma : X \to Y$ such that $S_Y \circ \sigma = \sigma \circ S_X$ where S_X, S_Y are the shift actions on X and Y respectively is called a *morphism*. A well-known Theorem of Curtis, Lyndon and Hedlund which can be found in full generality in [11] asserts that morphisms are equivalent to maps defined by local rules as follows: There exists a finite $F \subset G$ and $\Phi : A^F \to B$ such that $\forall x \in X : \sigma(x)_g = \Phi(S^{g^{-1}}(x)|_F)$. A surjective morphism is called a *factor map* and we denote the existence of a factor map from X to Y by $X \twoheadrightarrow Y$. A bijective morphism is called a *conjugacy* and the fact that two subshifts are conjugate is written $X \cong Y$.

Definition 19 *A subshift $X \subset A^G$ is of* finite type *or SFT if there exist a finite set $\mathcal{F} \subset L(A^G)$ of forbidden patterns such that $X = X_{\mathcal{F}}$. A subshift is* sofic *if it's the image of an SFT via a factor map.*

Figure 3: In the left we see for $n = 2$ a set of patterns which do not contain forbidden subpatterns. In the right the transformation of one of these patterns into a Wang tile.

Definition 20 *Let* S *be a set of generators for the group* G*. A subshift* $X \subset A^G$ *is said to be* nearest neighbor with respect to S *if there exists a set* $\mathcal{F} \subset L(A_G)$ *such that* $X = X_{\mathcal{F}}$ *and every pattern* $p \in \mathcal{F}$ *satisfies* $\mathrm{supp}(p) = \{1_G, s\}$ *for some* $s \in \mathsf{S}$*. Such a set of forbidden patterns is also said to be* nearest neighbor.

Nearest neighbor subshifts can be seen as colorings of the Cayley graph $\Gamma(G, \mathsf{S})$ such that for each edge (g, gs) the choices of color are restricted.

Example 18 *The set* $X = \{x \in A^G \mid \forall s \in \mathsf{S}, x_g \neq x_{gs}\}$ *is a nearest neighbor subshift.*

This notion also encompasses Wang tiles as studied in Section 2.1. The following example makes this explicit.

Example 19 *Consider* $\mathbb{Z}^2$ *with the set of generators* $\mathsf{S} = \{(1,0),(0,1)\}$*. Given a set of Wang tiles* τ *the set of all tilings of the plane by* τ *is a nearest neighbor subshift. Indeed, it corresponds to* $X_{\mathcal{F}} \subset \tau^{\mathbb{Z}^2}$ *where the patterns* $p \in \mathcal{F}$ *with support* $\{(0,0),(1,0)\}$ *(respectively* $\{(0,0),(1,0)\}$*) are exactly those such that* $(p_{(0,0)})_E \neq (p_{(1,0)})_W$ *(respectively* $(p_{(0,0)})_N \neq (p_{(0,1)})_S$*).*

Every nearest neighbor subshift is of finite type, indeed, any set $\mathcal{F}$ satisfying the constrains satisfies $\#(\mathcal{F}) \leq \#(A)^{2\#(\mathsf{S})}$. The converse is false. For instance, the sequence of $\mathbb{Z}$-subshifts $\{X_n\}_{n\in\mathbb{N}}$ where $X_n \subset \{0,1\}^{\mathbb{Z}}$ is defined by $\mathcal{F}_n = \{1^n\}$ is a countable set of subshifts of finite type which satisfy that $1^{n-1} \in L(X_n) \setminus \bigcup_{m<n} L(X_m)$. Therefore an infinite number of them are forcefully not nearest neighbor. Nevertheless, every subshift of finite type is conjugate to a nearest neighbor subshift.

Before showing that result in generality, we illustrate informally in Figure 3 how this conjugacy works in the case we would like to turn a $\mathbb{Z}^2$ subshift into an equivalent set of Wang tiles. As the set of forbidden patterns is finite, there exists a big enough $n \in \mathbb{N}$ such that the support of every forbidden pattern is contained in $[0,n]^2$. Then one can construct the set of colorings of $[0,n]^2$ which do not contain forbidden patterns and turn each one of them into Wang tiles which through their adjacency colors force two contiguous patterns to overlap. This technique gives a one to one correspondence between the set of valid tilings of the Wang tiles and the configurations in the original subshift which can be shown to be a conjugacy.

Proposition 20 *Every subshift of finite type is conjugate to a nearest neighbor subshift.*

Proof: Let $\mathcal{F}$ be a finite set of forbidden patterns defining $X_{\mathcal{F}} \subset A^G$ and let $N = \max_{p\in\mathcal{F}, g\in\mathrm{supp}(p)} |g|_{\mathsf{S}}$. We define the alphabet

$$B = \{\widetilde{p} \in A^{B_{\mathsf{S}}(1_G,N)} \mid \forall p \in \mathcal{F}, \widetilde{p}|_{\mathrm{supp}(p)} \neq p\}.$$

Consider the set of forbidden patterns $\mathcal{G}$ containing $q \in B^{\{1_G,s\}}$ if and only if $\exists g \in B_{\mathsf{S}}(1_G,N) \cap B_{\mathsf{S}}(s,N)$ such that $(q_{1_G})_g \neq (q_s)_{s^{-1}g}$. By definition $X_{\mathcal{G}}$ is nearest neighbor for S. We claim $X_{\mathcal{G}} \cong X_{\mathcal{F}}$. Indeed, consider the morphism $\sigma : X_{\mathcal{G}} \to X_{\mathcal{F}}$ given by $\sigma(y)_g = (y_g)_{1_G}$. Let y, z be two different configurations in $X_{\mathcal{G}}$. Modulo shifting these configurations we can suppose $y_{1_G} \neq z_{1_G}$, meaning there exists $h \in B_{\mathsf{S}}(1_G,N)$ such that $(y_{1_G})_h \neq (z_{1_G})_h$. Write $h =_G s_1, \cdots, s_n$ for some $n \leq N$ such that each $s_i \in \mathsf{S}$. The forbidden patterns of $\mathcal{G}$ force that $(y_{1_G})_h = (y_{s_1})_{s_1^{-1}h}$ and $(y_{s_1})_{s_1^{-1}h} = (y_{s_1 s_2})_{s_2^{-1}s_1^{-1}h}$ and so on we obtain:

$$(y_{1_G})_h = (y_{s_1,\cdots,s_n})_{s_n^{-1},\cdots,s_1^{-1}h} = (y_h)_{1_G}.$$

Similarly, $(z_{1_G})_h = (z_h)_{1_G}$, therefore $(y_h)_{1_G} \neq (z_h)_{1_G}$ and thus $\sigma(y)_h \neq \sigma(z)_h$ showing that σ is injective. Given $x \in X_{\mathcal{F}}$ we can define $y \in B^G$ given by $y_g = S^{g^{-1}}(x)|_{B_{\mathsf{S}}(1_G,N)}$. y satisfies $\sigma(y) = x$ and $y \in X_{\mathcal{G}}$ thus proving the surjectivity of σ. Hence σ is a conjugacy. ■

We remark that the subshift $X_{\mathcal{G}}$ constructed in the previous proof is called the *higher-block shift* of X and denoted by $X^{[N]}$ in dimension one [35].

3.2 Domino Problem

From their representation with a finite automaton [36], the existence of a configuration in a $\mathbb{Z}$-SFT is equivalent to the existence of a cycle in a finite labeled graph, which is decidable.

In $\mathbb{Z}^2$ the domino problem asks for an algorithm which receives as an input a set of Wang tiles and decides whether they admit a tiling of the plane. A similar problem in the context of subshift would be to take a set of forbidden patterns $\mathcal{F}$ and ask whether the subshift $X_{\mathcal{F}}$ is non-empty. These two problems, while defined in different settings do inherently refer to the same objects and are both undecidable as shown in Section 2.4.

In general groups these problems become more complex to define for two reasons. From the side of the domino problem we have to replace Wang tiles by nearest neighbor subshifts, which raises the question of which set of generators to use. From the side of the emptiness problem we need a way to code the set of forbidden patterns such that a Turing machine can interpret them.

3.2.1 Definitions

We start this section by giving formal definitions for the domino problem and the Emptiness problem, then we prove that the decidability status of these two problems are the same, and does not depend on the choice for the generating set of the group considered.

Definition 21 *Let* S *be a fixed set of generators for a group* G*. The* domino problem with respect to S *is defined as the set* $\mathtt{DP}(G, \mathsf{S})$ *of codings of nearest neighbor for* S *sets of forbidden patterns* $\mathcal{F}$ *such that* $X_{\mathcal{F}} \neq \emptyset$*.*

One way to formally code a nearest neighbor set of forbidden patterns is to identify each pattern as a triple in $A^2 \times S$ and identify A to a finite set of words in $\{0,1\}^*$. We say that the domino problem with respect to S is *decidable* if $\mathtt{DP}(G, \mathsf{S})$ is a decidable language.

In order to define the emptiness problem, we first need to describe how to code general patterns.

Definition 22 *Let* G *be a finitely generated group,* $S \subset G$ *a finite generating set and* A *a finite alphabet. A* pattern coding c *is a finite set of tuples* $c = \{(w_i, a_i)\}_{i \in I}$ *where* $w_i \in S^*$ *and* $a_i \in A$*. Given a set* $\mathcal{C}$ *of pattern codings we define the subshift* $X_{\mathcal{C}}$ *by:*

$$X_{\mathcal{C}} = A^G \setminus \bigcup_{g \in G, c \in \mathcal{C}} \bigcap_{(w,a) \in c} [a]_{gw}$$

Note that if a pattern coding does not represent an actual pattern, that is, if two words representing the same group element get paired with different letters then $\bigcap_{(w,a)\in c}[a]_{gw}$ is empty and the coding does not contribute at all in the formula above.

Definition 23 *Let* S *be a fixed set of generators for a group* G*. The* emptiness problem with respect to S *is defined as the set* $\mathtt{EP}(G, \mathsf{S})$ *of sets of pattern codings* $\mathcal{C}$ *such that* $X_{\mathcal{C}} \neq \emptyset$*.*

Using the same technique as in Proposition 16 we obtain that the computational properties of the emptiness problem are independent of the chosen set of generators. Therefore, analogously to the case of the word problem for groups, we can plainly speak about the emptiness problem for a given group $\mathtt{EP}(G)$ and use an arbitrary set of generators.

Proposition 21 *For every pair* S, S' *of finite set of generators of* G *we have that* $\mathtt{EP}(G, \mathsf{S})$ *is many-one equivalent to* $\mathtt{EP}(G, \mathsf{S}')$*.*

Proposition 22 *Let* S *be a finite set of generators of* G*. Then* $\mathtt{DP}(G, \mathsf{S})$ *is many-one equivalent to* $\mathtt{EP}(G, \mathsf{S})$*.*

Proof: Clearly $\texttt{DP}(G,\mathsf{S}) \leq_m \texttt{EP}(G,\mathsf{S})$ as any instance of $\texttt{DP}(G,\mathsf{S})$ is an instance of $\texttt{EP}(G,\mathsf{S})$. To prove the converse, we would like to use the conjugacy from Proposition 20 but the construction of the new alphabet might not be computable if the word problem of G is undecidable. We bypass this problem as follows. Given a set of pattern codings $\mathcal{C}$ we compute $N = \max_{c\in\mathcal{C}} \max_{(w_i,a_i)\in c} |w_i|$ and

$$B = \{b : \bigcup_{n\leq N} S^n \to A \mid \forall c \in \mathcal{C}, \exists (w,a) \in c : b_w \neq a\}$$

That is, the set of all colorings of words of length at most N such that no pattern coding from $\mathcal{C}$ appears. This set is computable, and the nearest neighbor set of forbidden patterns $\mathcal{G}$ containing $q \in B^{\{1_G,s\}}$ if and only if $\exists w \in \bigcup_{n\leq N-1} S^n$ such that $(q_{1_G})_{sw} \neq (q_s)_w$ also is. Therefore we obtain an instance of $\texttt{DP}(G,S)$.

If $X_{\mathcal{C}}$ is non-empty we can construct $y \in X_{\mathcal{G}}$ by setting $(y_g)_w = x_{gh}$ where $h =_G w$ is the group element coded by w. It clearly does not contain any coding from $\mathcal{C}$ by definition. Conversely, analogously to the proof of Proposition 20 we obtain that for every $y \in X_{\mathcal{G}}$, $g \in B_S(1_G, N)$ and $w \in \bigcup_{n\leq N} S^n$ such that $g =_G w$ then $(y_{1_G})_w = (y_g)_\epsilon$ where ϵ is the empty word. In particular we deduce that every symbol $b \in B$ appearing in a configuration must satisfy that $b_{w_1} = b_{w_2}$ for each $w_1 =_G w_2$. We can thus construct a configuration $x \in X_{\mathcal{C}}$ from $y \in X_{\mathcal{G}}$ defined as $x_g = (y_g)_\epsilon$. We conclude that $\texttt{EP}(G,\mathsf{S}) \leq_m \texttt{DP}(G,\mathsf{S})$ and thus $\texttt{DP}(G,\mathsf{S}) \equiv_m \texttt{EP}(G,\mathsf{S})$ ■

Mixing the two previous propositions, we get:

$$\texttt{DP}(G,\mathsf{S}) \equiv_m \texttt{EP}(G,\mathsf{S}) \equiv_m \texttt{EP}(G,\mathsf{S}') \equiv_m \texttt{DP}(G,\mathsf{S}')$$

Corollary 23 *Let S,S' be a finite set of generators of G. Then $\texttt{DP}(G,\mathsf{S})$ is many-one equivalent to $\texttt{DP}(G,\mathsf{S}')$.*

We can therefore just speak plainly about the *domino problem* $\texttt{DP}(G)$ of a group G as the domino problem with respect to any set of generators. The results of this sections give us the liberty to treat the domino problem in any of the previous formats, that is, using any finite set of generators, and either with nearest neighbor forbidden patterns or with sets of pattern codings.

3.2.2 Basic properties

Theorem 24 *For any group G then $\texttt{WP}(G) \leq_m \overline{\texttt{DP}(G)}$. In particular the domino problem is undecidable for any group with undecidable word problem.*

Proof: More precisely, we are going to show $\texttt{WP}(G) \leq_m \overline{\texttt{EP}(G)}$. Consider the alphabet $A = \{0,1,2\}$. Given $w \in \mathsf{S}^*$ an input of the word problem we associate the set of pattern codings $\mathcal{C} = \{c_0, c_1, c_2\}$ where $c_i = \{(\epsilon, i), (w, i)\}$. This set $\mathcal{C}$ is clearly computable from w. In other words, the set $\mathcal{C}$ forces the symbol in each group element g to be different from the one in gw.

If $w \in \texttt{WP}(G)$ then $w =_G 1_G$, therefore $[i]_\epsilon \cap [i]_w = [i]_{1_G}$ and so $X_{\mathcal{C}} \cap [i]_w = \emptyset$ for each $i \in \{0,1,2\}$. We deduce that

$$\emptyset = \bigcup_{i\in\{0,1,2\}} X_{\mathcal{C}} \cap [i]_w = X_{\mathcal{C}} \cap A^G = X_{\mathcal{C}} \text{ and thus } \mathcal{C} \in \overline{\texttt{EP}(G)}.$$

In the case where $w \neq_G 1_G$ we show that $X_{\mathcal{C}} \neq \emptyset$. Indeed, let $w =_G g \in G$. As $g \neq 1_G$ then $\langle g\rangle$ is a non-trivial cyclic subgroup. So either $\langle g\rangle \cong \mathbb{Z}$ or $\langle g\rangle \cong \mathbb{Z}/n\mathbb{Z}$ for some $n \geq 2$. We construct $y \in A^{\langle g\rangle}$ differently for each case as follows: In the case $\langle g\rangle \cong \mathbb{Z}$ we set $y_{g^m} = m \mod 2$. In the case $\langle g\rangle \cong \mathbb{Z}/n\mathbb{Z}$ we distinguish again two cases, if n is even then we set $y_{g^m} = m \mod 2$. Otherwise we just set $y_{g^m} = m \mod 2$ if $n \nmid m$, otherwise $y_{g^m} = 2$. One can verify that in each case $\forall h \in \langle g\rangle$ $y \notin [i]_h \cap [i]_{hg}$.

Consider a set of left representatives L for the quotient $G/\langle g\rangle$. We can define $x \in A^G$ by $x_{\ell h} = y_h$ for every $\ell \in L$ and $h \in \langle g\rangle$. By definition we have for each ℓ, h and $i \in A$ then $x \notin [i]_{\ell h} \cap [i]_{\ell hw}$ and thus $x \in X_{\mathcal{C}}$ and hence $X_{\mathcal{C}} \neq \emptyset$. ■

Fix a group G, a finite generating set S, a finite alphabet and a set of codings of nearest neighbor forbidden patterns $\mathcal{F}$. Suppose $X_{\mathcal{F}}$ is empty. Then by compactness, there exists a

size N such that the ball of size $B_{\mathsf{S}}(1_G, N)$ fails to be colored without patterns from $\mathcal{F}$. So a naive procedure, that would consist in exhaustively searching for a valid coloring of balls of increasing size, will eventually stop because such a coloring does not exist for the ball of size N. This only restriction we need to perform such a procedure is that the group structure should be enumerable, which in formalized in the following proposition.

Proposition 25 *The domino problem is co-recursively enumerable for any recursively presented group.*

Proof: As G is recursively presented, there is a Turing machine $\mathcal{M}_1$ which on input $w \in \mathsf{S}^*$ returns `YES` if and only if $w =_G 1_G$. Consider the following algorithm $\mathcal{M}_2$:

1. Initialize $n \leftarrow 1$.
2. Do the following procedure:
   - For each pair of words $u, v \in \mathsf{S}^*$ of length at most n. Run n steps $\mathcal{M}_1$ on entry uv^{-1}.
   - For each $m \in \{1, \cdots, n\}$ construct the set X_m of functions $p : \bigcup_{k \leq m} \mathsf{S}^k \to A$ such that $p_u = p_v$ for each pair (u, v) where $\mathcal{M}$ answered `YES` and where no forbidden pattern appears.
3. If some X_m is empty, return `YES`. Otherwise do $n \leftarrow n + 1$ and go to 2.

The previous algorithm answers `YES` if and only if the instance of $\mathtt{DP}(G)$ generates an empty subshift. Indeed, if the subshift X is non-empty then we can take $x \in X$ and define $p \in X_m$ as $p_w = x_w$. Conversely if $X \subset A^G$ is empty there exists $N \in \mathbb{N}$ such that every $p \in A^{B_{\mathsf{S}}(1_G, N)}$ satisfies $X \cap [p] = \emptyset$. Otherwise we may choose $x_n \in [p_n] \cap X$ where $p_n \in A^{B_{\mathsf{S}}(1_G, n)}$ and any accumulation point of $\{x_n\}_{n \in \mathbb{N}}$ would be in X. Therefore it suffices to run the procedure for sufficient steps such that every pair (u, v) of length at most N such that $u =_G v$ is identified and $\mathcal{M}_2$ will forcefully obtain that $X_N = \emptyset$ and answer `YES`. ■

In other words, Proposition 25 means that as soon as the group is recursively presented, the difficult part of the domino problem is to detect if a valid tiling exists.

3.3 Inheritance properties

Proposition 26 *For every finitely generated $H \leq G$ we have $\mathtt{DP}(H) \leq_m \mathtt{DP}(G)$.*

Proof: Let S_H and S_G be sets of generators for H and G respectively. As $H \leq G$ then $\mathsf{S}_G \cup \mathsf{S}_H$ also generates G. Any input of $\mathtt{DP}(H, \mathsf{S}_H)$ is also an input of $\mathtt{DP}(G, \mathsf{S}_H \cup \mathsf{S}_G)$. If the original input produces an empty subshift, then it also does so in the image as the subgroup $H \leq G$ admits no valid configuration. Conversely, if the original input admits a configuration, then it can be used to tile each lateral class G/H as in the proof of Theorem 24 and therefore the subshift produced by the image is also non-empty. ■

From Theorem 11 and Proposition 26 we get:

Corollary 27 *If $\mathbb{Z}^2$ embeds into G then $\mathtt{DP}(G)$ is undecidable.*

Proposition 28 *For every finitely generated normal subgroup $H \trianglelefteq G$ we have $\mathtt{DP}(G/H) \leq_m \mathtt{DP}(G)$.*

Proof: Every quotient of a finitely generated group is finitely generated so $\mathtt{DP}(G/H)$ is well defined. Let L be a set of representatives of G/H in G and let $\eta : G/H \to L$ be this identification. Consider finite sets $\mathsf{S}_{G/H}$, S_H of generators of G/H and H respectively and let $\mathsf{S}_L = \eta(\mathsf{S}_{G/H})$. We remark that if $f_1, f_2 \in G/H$ then $\eta(f_1 f_2) = \eta(f_1)\eta(f_2)h$ for some $h \in H$. In particular as every $g \in G$ can be written as $g = \ell h$ for some $\ell \in L$ and $h \in H$ we obtain that each $g \in G$ can be written as uv where $u \in \mathsf{S}_L$ and $v \in \mathsf{S}_H$. Therefore $\mathsf{S}_L \cup \mathsf{S}_H$ generate G.

Consider an instance $\mathcal{F}$ of $\mathtt{DP}(G/H, \mathsf{S}_{G/H})$. We construct an instance $\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2$ of $\mathtt{DP}(G, \mathsf{S}_L \cup \mathsf{S}_H)$ such that $X_{\mathcal{G}} = \emptyset$ if and only if $X_{\mathcal{F}} = \emptyset$. For each pattern with support $\{1_{G/H}, r\}$ with $r \in \mathsf{S}_{G/H}$ in the original instance we add the same pattern with support $\{1_G, \eta(r)\}$ in $\mathcal{G}_1$. We construct $\mathcal{G}_2$ as the set of all patterns p with support $\{1_G, s\}$ for $s \in \mathsf{S}_H$ such that $p_{1_G} \neq p_s$. Namely $\mathcal{G}_1$ copies the original rules in every quotient and $\mathcal{G}_2$ forces that

$\forall y \in X_{\mathcal{G}}$ every configuration is invariant by translations by H. This construction of $\mathcal{G}$ is computable and gives an instance of $\mathtt{DP}(G, \mathsf{S}_L \cup \mathsf{S}_H)$.

Suppose $X_{\mathcal{F}} \subset A^{G/H}$ is non-empty. Then from $x \in X_{\mathcal{F}}$ we can construct $y \in A^G$ defined by $y_{\ell h} = x_{\eta^{-1}(\ell)}$. By definition we have that for each $g \in G$ and $s \in \mathsf{S}_H$ then $y_g = y_{gs}$ and so no pattern from $\mathcal{G}_2$ appears. Also, given $r \in \mathsf{S}_L$ we have

$$\begin{aligned} y_{\ell hr} &= y_{\ell r(r^{-1}hr)} \\ &= y_{(\ell r)h'} \text{ for some } h' \in H \text{ as } H \trianglelefteq G \\ &= y_{\ell' h'' h'} \text{ for some } h'' \in H \\ &= y_{\ell'} \end{aligned}$$

Where $\ell' = \eta(\eta^{-1}(\ell)\eta^{-1}(r))$. Therefore $y_{\ell hr} = x_{\eta^{-1}(\ell)\eta^{-1}(r)}$ meaning that no patterns from $\mathcal{G}_1$ appear. Therefore $y \in X_{\mathcal{G}}$.

Conversely let $y \in X_{\mathcal{G}}$ and consider $x \in A^{G/H}$ defined by $x_g = y_{\eta(g)}$. Suppose a forbidden pattern with support $\{1_{G/H}, r\}$ from $\mathcal{F}$ appears in x in position g. Therefore $y_{\eta(gr)} = y_{\eta(g)\eta(r)h} = y_{\eta(g)\eta(r)}$ for some $h \in H$ and thus the same forbidden pattern appears in y with support $\{1_G, \eta(r)\}$. This implies that $x \in X_{\mathcal{F}} \neq \emptyset$. ■

Proposition 29 *Let $H \leq G$ such that $[G : H] < \infty$. Then $\mathtt{DP}(G) \equiv_m \mathtt{DP}(H)$.*

Proof: The direction $\mathtt{DP}(H) \leq_m \mathtt{DP}(G)$ is direct from Proposition 26. Conversely, to prove $\mathtt{DP}(G) \leq_m \mathtt{DP}(H)$ we can suppose that $H \trianglelefteq G$. Indeed, if H is not normal we can find $N \leq H$ such that $N \trianglelefteq G$ and $[G : N] < \infty$ (see Exercise 9). If we prove that $\mathtt{DP}(G) \leq_m \mathtt{DP}(N)$ we would have $\mathtt{DP}(G) \leq_m \mathtt{DP}(N) \leq_m \mathtt{DP}(H)$ and thus $\mathtt{DP}(G) \leq_m \mathtt{DP}(H)$.

Let $X \subset A^G$ be a subshift and R a set of representatives of the right lateral classes $G \backslash H$ which contains 1_G. We define the *R-higher power shift* of X as the set

$$X^{[R]} := \{y \in (A^R)^H \mid \exists x \in X, \forall (h, r) \in H \times R, (y_h)_r = x_{hr}\}.$$

The set $X^{[R]}$ is indeed an H-subshift and $X = \emptyset \iff X^{[R]} = \emptyset$. As every finite index subgroup of a finitely generated group is itself finitely generated (see Exercise 10) we can take a set of generators S_H for H and thus $\mathsf{S}_H \cup R$ is a finite set of generators for G. Let $D = \mathsf{S}_H \cup (RRR^{-1} \cap H)$ and $E = RDR^{-1}$. Note that as both R and S_H are finite then E also is. Furthermore as $1_G \in R$ then $\mathsf{S}_H \subset E$ and as $H \trianglelefteq G$ we have $E \subset H$, therefore $H = \langle E \rangle$. Given an instance $\mathcal{F}$ of $\mathtt{DP}(G, \mathsf{S}_H \cup R)$ with alphabet A we are going to construct an instance $\mathcal{G}$ of $\mathtt{DP}(H, E)$ with alphabet A^R such that $X_{\mathcal{G}} = X_{\mathcal{F}}^{[R]}$ as follows: for every pattern $p \in \mathcal{F}$ with $\mathrm{supp}(p) = \{1_G, s\}$ with $s \in \mathsf{S}_H$ and $r \in R$ we put in $\mathcal{G}$ all the patterns q with support $\{1_H, rsr^{-1}\}$ such that $(q_{1_H})_r = p_{1_G}$ and $(q_{rsr^{-1}})_r = p_s$. This will take care of all patterns with support $\{1_G, s\}$ and $s \in \mathsf{S}_H$. For the remaining patterns let $(a, b) \in R^2$. By definition it is always possible to write $ab = \bar{h}c$ for some $c \in R$ and some $\bar{h} \in RRR^{-1} \cap H$. Now for every pattern $p \in \mathcal{F}$ with $\mathrm{supp}(p) = \{1_G, b\}$ with $b \in R$ and $a \in R$ we let $\bar{h} \in RRR^{-1} \cap H$ such that $ab = \bar{h}c$ and we add to $\mathcal{G}$ all patterns q with support $\{1_H, a^{-1}\bar{h}\}$ such that $(q_{1_H})_a = p_{1_G}$ and $(q_{a^{-1}\bar{h}})_c = p_b$.

Some of the patterns in $\mathcal{G}$ defined above will have some trivial support, in this case we just consider that as a restriction on the alphabet A^R. Clearly as R is fixed beforehand this construction can be computed from an instance of $\mathtt{DP}(G, \mathsf{S}_H \cup R)$. We leave as an exercise to the reader to verify that $X_{\mathcal{G}} = X_{\mathcal{F}}^{[R]}$ and thus conclude that $\mathtt{DP}(G) \leq_m \mathtt{DP}(H)$. ■

We say two groups G_1, G_2 are commensurable if they contain finite index subgroups $H_1 \leq G_1$ and $H_2 \leq G_2$ such that $H_1 \cong H_2$.

Corollary 30 *Let G_1, G_2 be two commensurable groups, then $\mathtt{DP}(G_1) \equiv_m \mathtt{DP}(G_2)$. Said otherwise, the domino problem is an invariant of commensurability.*

3.4 Classes of groups

3.4.1 Virtually free groups

Proposition 31 *Let F be a free group of finite rank. Then $\mathtt{DP}(F)$ is decidable.*

Proof: Let $n = \mathrm{rank}(F)$ and let $S = \{s_1, \dots, s_n, s_1^{-1}, \cdots, s_n^{-1}\}$ be the set of free generators of F. Consider an instance $\mathcal{F}$ of $\mathtt{DP}(F, S)$ over an alphabet A. We say a symbol $a \in A$ is *extensible with respect to* $B \subset A$ if for every $s \in S$ there exists $b \in B$ such that neither of the patterns p, q with supports $\mathrm{supp}(p) = \{1_F, s\}$ and $\mathrm{supp}(p) = \{1_F, s^{-1}\}$ defined by $p_{1_F} = a$, $p_s = b$, $q_{1_F} = b$, $q_{s^{-1}} = a$ belong to $\mathcal{F}$. Said otherwise, a is extensible with respect to B if for every direction $s \in S$ it's possible to put an a next to some $b \in B$ in position s without creating a forbidden pattern.

Consider the Turing machine $\mathcal{M}$ which receives an instance of $\mathtt{DP}(F, S)$ and does the following:

1. Initialize $E \leftarrow A$
2. Let E' be the subset of symbols of E which are extensible with respect to E.
3. If $E' \neq E$ assign $E \leftarrow E'$ and go to step 2.
4. If $E \neq \emptyset$ answer `YES`. Otherwise answer `NO`.

This procedure always ends in at most $|A|$ iterations of step 2. Clearly non-extensible symbols cannot appear in a configuration of $X_{\mathcal{F}}$. We deduce therefore that $X_{\mathcal{F}} \subset E^F$ at any step of the algorithm. This implies that if $\mathcal{M}$ answers `NO` then indeed $X_{\mathcal{F}} = \emptyset$. If $\mathcal{M}$ answers `YES` then E stabilizes into a non-empty set of extensible with respect to E symbols. Fix for every $a \in E$ a function $\varphi_a : S \to E$ which gives an symbol in E which can be put next to a in direction s. We define $x \in E^F$ inductively as follows: Fix $x_{1_G} = a \in E$. Suppose x is defined over all words $w \in S^*$ of length $|w| \leq n$. For each non-reducible word ws we let $x_{ws} = \varphi_{(x_w)}(s)$. As the Cayley graph of F is a $2n$-regular infinite tree this construction does not generate any forbidden patterns and hence $x \in X_{\mathcal{F}}$. ■

Definition 24 *Let $\mathcal{P}$ be a group property. A group is said to be virtually $\mathcal{P}$ if it contains a finite index subgroup which satisfies such property.*

Integrating the previous proposition with Proposition 29 we obtain the following theorem.

Theorem 32 *Every virtually free group has decidable word problem.*

We would like to remark a nice application of Proposition 31. If G is finitely generated by some finite set S it admits a presentation $G \cong \langle \mathsf{S} | \mathsf{R} \rangle = F_{\mathsf{S}}/N_{\mathsf{R}}$ where $N_{\mathsf{R}} \trianglelefteq F_{\mathsf{S}}$. As $\mathtt{DP}(F_{\mathsf{S}})$ is decidable, Proposition 28 implies that if N_{R} is finitely generated then $\mathtt{DP}(G)$ is decidable. If we put this together with the Nielsen-Schreier [37] theorem which states that every subgroup of a free group is itself free, we can write it in the following way.

Corollary 33 *Let (S, R) be a group presentation. If $\mathtt{DP}(\langle \mathsf{S} | \mathsf{R} \rangle)$ is undecidable then the free group N_{R} generated by the conjugate closure of R has infinite rank.*

Example 34 *Let $[a, b] = aba^{-1}b^{-1}$ denote the* commutator *of a and b. Let $\mathbb{Z}^2 \cong \langle a, b \mid [a, b] \rangle$. As $\mathtt{DP}(\mathbb{Z}^2)$ is undecidable, then $N_{[a,b]}$ has infinite rank. One can also easily verify that $N_{[a,b]} = [F, F] = \{[g, h] \mid g, h \in F\}$. This constitutes a new (and algebraic topology free) proof of the classical result stating that the commutator subgroup of a free group of rank 2 has infinite rank.*

3.4.2 Polycyclic groups

The class of polycyclic groups is one of the largest for which we can obtain a complete classification concerning the undecidability of the domino problem. This is achieved through the use of the properties proved in section 3.3. Polycyclic groups have indeed a lot of nice properties due to the fact that this is one of the largest classes of groups which is closed under subgroups and quotients, and that contain only finitely presented groups with decidable word problem. See [51] and [34] for more details on polycyclic groups.

Polycyclic groups are the solvable groups for which every subgroup is finitely generated. The best way to give examples of polycyclic groups is by the Auslander-Swan theorem:

Theorem 35 *Polycyclic groups are precisely solvable subgroups of $GL_n(\mathbb{Z})$.*

See [51, Chapter 5, Theorem 5] or [34, section 3.3] for a proof. By Tits Alternative [55], we therefore obtain that virtually polycyclic groups are precisely the subgroups of $GL_n(\mathbb{Z})$ that do not contain non-abelian free groups, or equivalently amenable subgroups of $GL_n(\mathbb{Z})$.

Polycyclic groups form a nice class of groups due to their many closure properties:

Proposition 36 *Quotients and subgroups of polycyclic groups are polycyclic. In particular, subgroups of polycyclic groups are always finitely generated.*

This property opens the possibility to do inductive proofs on polycyclic groups. This is done formally with the concept of the *Hirsch number*. The Hirsch number $h(G)$ of a polycyclic group G is the number of infinite factors in a series with cyclic or finite factors. The Hirsch number is always finite, and subgroups and quotients have a smaller Hirsch number than the group. More precisely:

Proposition 37

- *If G_1 is a subgroup of G_2, then $h(G_1) \leq h(G_2)$.*
- *If H is a normal subgroup of G, then $h(G) = h(G/H) + h(H)$*
- *$h(G) = 0$ iff G is finite*
- *$h(G) = 1$ iff G is virtually $\mathbb{Z}$*
- *$h(G) = 2$ iff G is virtually $\mathbb{Z}^2$.*

See in particular [51, Chapter 1.C].

We now are ready for the main theorem of this section

Theorem 38 *Let G be a virtually polycyclic group. Then G has an undecidable domino problem if and only if G is not virtually cyclic.*

Proof: One direction is clear. By Proposition 29, it is sufficient to prove the result for polycyclic groups.

We prove the result by induction on the Hirsch number. The result is clear for Hirsch number $0, 1, 2$. Now let G be a group of Hirsch number no less than 3.

It is known that every polycyclic group admit a non-trivial normal free abelian subgroup [51, Chapter 1, lemma 8].

Let H be such a subgroup. If $H = \mathbb{Z}^n$ for some $n > 2$, then H has an undecidable domino problem, and therefore G also has an undecidable domino problem by Proposition 26. Otherwise $H = \mathbb{Z}$. Then G/H is a polycyclic subgroup of Hirsch number $h(G) - 1 \geq 2$ and therefore has an undecidable domino problem. We conclude again by Proposition 28 that G has an undecidable domino problem. ■.

To which extent this theorem can be extended is open. Of course any group that contains a polycyclic group of Hirsch number greater than 2 also has an undecidable domino problem.

A natural direction is extending the theorem to all finitely generated solvable groups. How to do this is unclear. First, there are finitely generated solvable groups with an undecidable word problem. Furthermore, some of them do not contain a copy of $\mathbb{Z}^2$ which means the previous method cannot work. Examples include the Lamplighter groups and the Baumslag-Solitar groups. Baumslag-Solitar groups will be treated in the next section. Whether the Lamplighter group admits an undecidable domino problem remains open.

3.4.3 Baumslag-Solitar groups

In this section we prove the undecidability of the domino problem on Baumslag-Solitar groups. Given two non-zero integers m and n, we define $\mathbf{BS}(m, n)$ the *Baumslag-Solitar group of order* (m, n) as the two-generators and one-relator group with presentation

$$\mathbf{BS}(m, n) =< a, b | a^m b = b a^n > .$$

In particular $\mathbf{BS}(1, 1)$ is isomorphic to $\mathbb{Z}^2$. Since $\mathbf{BS}(-m, -n)$ is isomorphic to $\mathbf{BS}(m, n)$, it is enough to consider groups with $m > 0$. For simplicity, we also assume that $n > 0$. The case $n < 0$ is analogous.

We first discuss $\Gamma_{m,n}$, the Cayley graph of $\mathbf{BS}(m, n)$ for $m, n > 0$. Since $\mathbf{BS}(m, n)$ has two generators, every vertex in its Cayley graph $\Gamma_{m,n}$ has in-degree and out-degree 2.

The level associated with g of the Cayley graph $\Gamma_{m,n}$ is the induced subgraph obtained by keeping only the vertices of the coset $g\langle a\rangle = \{g.a^k : k \in \mathbb{Z}\}$. We denote it by $\mathcal{L}_g$ and we say that the vertex g defines the level $\mathcal{L}_g$. The level $\mathcal{L}_{gb}$ is a predecessor of the level $\mathcal{L}_g$, while the latter is a successor of the former, for all group elements g. Note that each level has m predecessors and n successors.

Our tilings will be colorings of the edges of the Cayley graph $\Gamma_{m,n}$. The local constraint is given in terms of a set of allowed patterns on the edges

$$\varepsilon \longrightarrow a \longrightarrow a^2 \longrightarrow \cdots \longrightarrow a^m \longrightarrow a^m b = ba^n$$

and

$$\varepsilon \longrightarrow b \longrightarrow ba \longrightarrow ba^2 \longrightarrow \cdots \longrightarrow ba^n = a^m b,$$

see the left side of Figure 4 for the case $m = 3, n = 2$. For each group element g the pattern of this shape found at position g must be among the allowed patterns.

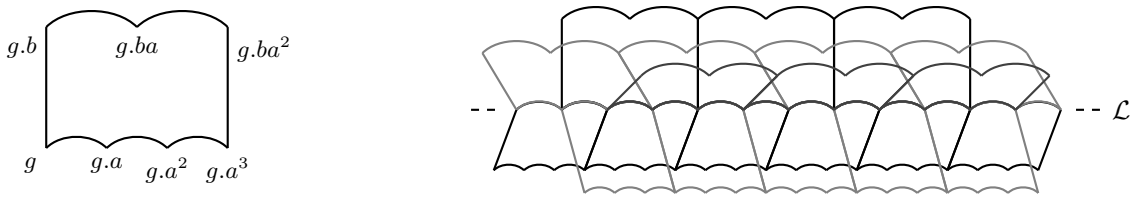


Figure 4: One the left: the shape of the tiles in $\Gamma_{3,2}$. On the right: some levels in $\Gamma_{3,2}$. The level $\mathcal{L}$ has two successor levels (drawn below the level) and three predecessor levels (drawn above it).

The Cayley graph $\Gamma_{m,n}$ can be projected into the Euclidean plane by a function $\Phi_{m,n} : \mathbf{BS}(m,n)1\mathbb{R}^2$, defined as follows. Let w be a finite word on the alphabet $A = \{a, b, a^{-1}, b^{-1}\}$. Then any element of $\mathbf{BS}(m,n)$ can be represented by such a word, but this representation is of course non unique. If x is a letter of A, we denote by $|w|_x$ the number of occurrences of x in the word w. We then define for $x \in A$ the contribution of x to w by $\| w \|_x = |w|_x - |w|_{x^{-1}}$.

Let $\psi_{m,n} : A^*1\mathbb{R}$ be the function defined by induction on the length of the word by

$$\begin{cases} \psi_{m,n}(\varepsilon) = 0 \text{ where } \varepsilon \text{ is the empty word} \\ \psi_{m,n}(w.b) = \psi_{m,n}(w.b^{-1}) = \psi_{m,n}(w) \\ \psi_{m,n}(w.a) = \psi_{m,n}(w) + \left(\frac{m}{n}\right)^{\|w\|_b} \\ \psi_{m,n}(w.a^{-1}) = \psi_{m,n}(w) - \left(\frac{m}{n}\right)^{\|w\|_b} \end{cases}$$

Lemma 39 *For every $u, v \in A^*$ one has*

$$\psi_{m,n}(u.v) = \psi_{m,n}(u) + \left(\frac{m}{n}\right)^{\|u\|_b} \psi_{m,n}(v).$$

Proof: By induction on the length of v. ■

We can now define the projection function $\Phi_{m,n} : \mathbf{BS}(m,n) \to \mathbb{R}^2$ which associates to every element g of $\mathbf{BS}(m,n)$ its coordinates on the Euclidean plane:

$$\Phi_{m,n}(g) = \left(\psi_{m,n}(w), \| w \|_{b^{-1}}\right),$$

where w is a word representing g. The following proposition states that the definition does not depend on the choice of w. Its proof is a simple application of Lemma 39.

Proposition 40 *The function $\Phi_{m,n}$ is well defined on $\mathbf{BS}(m,n)$.*

For every element $g \in \mathbf{BS}(m,n)$, define the shift of g as the first coordinate of $\Phi_{m,n}(g)$, that takes rational values, and the height of g as the second coordinate of $\Phi_{m,n}(g)$, that takes integer values.

All elements belonging to the same level project on the same horizontal line, thus we can speak of the height of a level. The height is $\| w \|_{b^{-1}}$ for the words w that represent the elements of the level.

Remark 41 *The function $\Phi_{m,n}$ is not injective. Let $m = 3$ and $n = 2$. Consider the word*

$$\omega = bab^{-1}a^2ba^{-1}b^{-1}a^{-2}.$$

We have

$$\Phi_{3,2}(\omega) = \Phi_{3,2}(\varepsilon) = (0, 0).$$

However, freely reduced words that do not contain $b^{-1}a^{km}b$ or $ba^{kn}nb^{-1}$ as subwords, for any integer k, cannot represent the identity in $\mathbf{BS}(m, n)$. Thus ω and ε represent different elements of the group. Moreover, Baumslag-Solitar groups are HNN-extensions of $\mathbb{Z}$, thus from Britton's lemma it follows that a finite subgroup of Baumslag-Solitar group is conjugate to a finite subgroup of $\mathbb{Z}$. Since ω is not the identity, it has infinite order in $\mathbf{BS}(3, 2)$. We see that there is an infinite cyclic subgroup that is projected by $\Phi_{3,2}$ to point $(0, 0)$. This will not be a problem in the sequel: the tile associated with an element $g \in \mathbf{BS}(m, n)$ will depend only on $\Phi_{m,n}(g)$.

Following the ideas from Section 2.5, we can construct a tile set such that if a level in a tiling of $\Gamma_{m,n}$ represents some $\overrightarrow{x} \in \mathbb{R}^2$, then its successor levels represent $f(\overrightarrow{x})$ where $f : \mathbb{R}^2 \to \mathbb{R}^2$ is a rational affine map. Going from one level to one of its successor level corresponds to one iteration of f, and a decrease of the height of the level by 1.

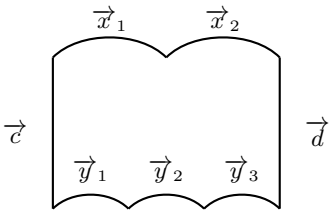


Figure 5: The general form of tiles in $\mathbf{BS}(3, 2)$.

Consider the case $\mathbf{BS}(3, 2)$. The tiles are of the form shown in Figure 5. We say that the tiles compute the function f if the relation

$$\frac{f(\overrightarrow{x}_1 + \overrightarrow{x}_2)}{2} + \overrightarrow{c} = \frac{\overrightarrow{y}_1 + \overrightarrow{y}_2 + \overrightarrow{y}_3}{3} + \overrightarrow{d}. \tag{1}$$

is satisfied.

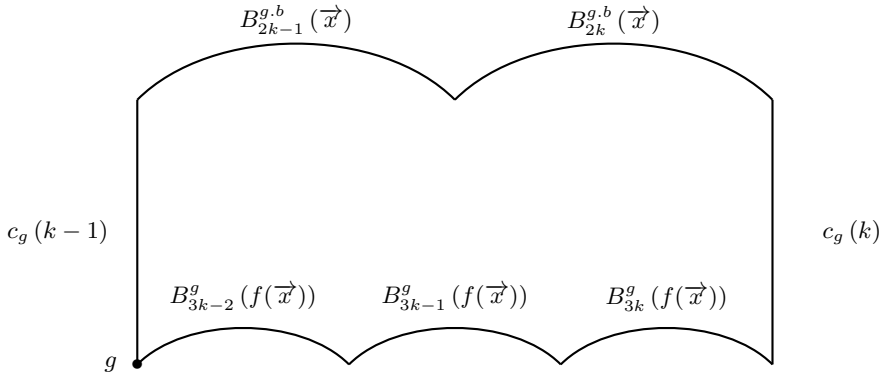
Consider a sequence of k such tiles on some level, next to each other so that the left vertical edge of a tile is the same as the right edge of the previous tile. Averaging (1) over all k tiles yields then

$$f(\overrightarrow{x}) + \frac{\overrightarrow{c}_1}{k} = \overrightarrow{y} + \frac{\overrightarrow{d}_k}{k}$$

where $\overrightarrow{x}$ is the average of the labels on the segment of $2k$ edges on the previous level, $\overrightarrow{y}$ is the average of the labels on the corresponding segment of $3k$ edges on the level below, and $\overrightarrow{c}_1$ and $\overrightarrow{d}_k$ are the left and right vertical edges of the first and the last tile in the row, respectively. Letting k grow to infinity, we see that if the previous level represents some $\overrightarrow{x} \in \mathbb{R}^2$ then the next level necessarily represents $f(\overrightarrow{x})$, as required.

For all $g \in \mathbf{BS}(3, 2)$ with $\Phi_{m,n}(g) = (\alpha, \beta)$ and for every $k \in \mathbb{Z}$, we define the translated balanced representation of $\overrightarrow{x}$ as the bi-infinite sequence $B^g(\overrightarrow{x}) = (B^g_k(\overrightarrow{x}))_{k\in\mathbb{Z}}$, where

$$B^g_k(\overrightarrow{x}) = \left\lfloor \left(\left(\frac{m}{n}\right)^{\beta} \alpha + k \right) \overrightarrow{x} \right\rfloor - \left\lfloor \left(\left(\frac{m}{n}\right)^{\beta} \alpha + (k-1) \right) \overrightarrow{x} \right\rfloor.$$



$$\text{with } c_g(k) = \tfrac{1}{2} f\left(\left\lfloor \left(\left(\tfrac{3}{2}\right)^{\beta-1}\alpha + 2k\right)\overrightarrow{x}\right\rfloor\right) - \tfrac{1}{3}\left\lfloor \left(\left(\tfrac{3}{2}\right)^{\beta}\alpha + 3k\right) f(\overrightarrow{x})\right\rfloor + k\overrightarrow{b}$$

For the same reasons as those invoked in Section 2.5 (the domain U is bounded, and the function f_i has rational coefficients), any rational function $f : U \subset \mathbb{R}^2 \to \mathbb{R}^2$ can be encoded by a finite tile set, so that for a given $\overrightarrow{x} \in U$, one can tile a level of $\mathbf{BS}(3,2)$ such that the balanced representations of $\overrightarrow{x}$ and $f(\overrightarrow{x})$ appear on the top and bottom labels respectively. We deduce the undecidability of the domino problem for $\mathbf{BS}(3,2)$. The proof for other Baumslag-Solitar groups is similar.

Theorem 42 *The domino problem is undecidable on Baumslag-Solitar groups.*

3.4.4 Groups $G_1 \times G_2$

In this section, we will prove the undecidability of the domino problem for groups that are direct products of infinite groups, i.e. groups of the form $G_1 \times G_2$, where G_1 and G_2 are infinite and finitely generated.

Of course, the only interesting case is when at least one of the two groups is an infinite torsion group, i.e. an infinite group where all elements are of finite order. Indeed, if it is not the case, then both G_1 and G_2 contain $\mathbb{Z}$ as a subgroup, and therefore $G_1 \times G_2$ contains $\mathbb{Z}^2$ as a subgroup, and thus has an undecidable domino problem by Proposition 26.

However, even if G does not contain a copy of $\mathbb{Z}$, it is still true that the Cayley Graph of G, as any infinite connected graph, contains infinitely many (undirected) paths. In fact, it can even been proven that some Cayley Graph of G (for a suitable choice of generators) can be covered by such paths. This is the purpose of the following theorem:

Theorem 43 ([53]) *Let G be an infinite, finitely generated group. Then there exists a finite set S s.t. the Cayley graph $\Gamma(G,\mathsf{S})$ of G with S as generators can be covered by disjoint biinfinite paths.*

This is a deep theorem with a non-trivial proof, see [53] for more details.

An equivalent way to say is as follows:

Theorem 44 *Let G be an infinite, finitely generated group. Then there exists a finite set S and a map $\mathfrak{next} : G \to G$, where $\mathfrak{next}(g)$ states which element of G is the next one in the biinfinite path g is lying on and that satisfies the following conditions:*

- *$\mathfrak{next}$ is one-to-one and onto. Its inverse will be called $\mathfrak{prev}$.*
- *$\mathfrak{next}(g)$ is a neighbor of g in $\Gamma(G,\mathsf{S})$. That is, for all g, $g^{-1}\mathfrak{next}(g) \in \mathsf{S}$*
- *Each path is infinite: for all $n > 0$ and all $g \in G$, $\mathfrak{next}^n(g) \neq g$.*

The last condition can be reformulated as follows: There is a map $\mathfrak{h}$ from G to $\mathbb{Z}$ s.t. $\mathfrak{h}(\mathfrak{next}(g)) = \mathfrak{h}(g) + 1$.

In all that follows, we suppose without loss of generality that S is symmetric, which means that additionally for all g, $g^{-1}\mathfrak{prev}(g) \in \mathsf{S}$.

Under these conditions, $\mathfrak{next}$ and $\mathfrak{prev}$ can be defined *locally*. Indeed, let $\mathfrak{n}(g) = g^{-1}\mathfrak{next}(g)$ and $\mathfrak{p}(g) = g^{-1}\mathfrak{prev}(g)$. Then $\mathfrak{n}$ and $\mathfrak{p}$ have values in the finite set S. Furthermore they satisfy the following two properties:

- If $\mathfrak{n}(g) = s$ then $\mathfrak{p}(gs) = s^{-1}$
- If $\mathfrak{p}(g) = s$ then $\mathfrak{n}(gs) = s^{-1}$

$\mathfrak{n}$ and $\mathfrak{p}$ can be interpreted in the Cayley Graph $\Gamma(G, \mathsf{S})$. At each vertex g of the Cayley Graph, two arrows are poited by. One of them correspond to $\mathfrak{n}(g)$, the other to $\mathfrak{p}(g)$. The first condition above states that if we start from some vertex g, follow the arrow pointed by $\mathfrak{n}$, and then at $g\mathfrak{n}(g)$ follow the arrow pointed by $\mathfrak{p}$ we come back to g.

Definition 25 *A* valid pair *for* (G, S) *is a pair* (n, p) *of maps from* G *to* S *s.t.*

- *If* $\mathsf{n}(g) = s$ *then* $\mathsf{p}(gs) = s^{-1}$.
- *If* $\mathsf{p}(g) = s$ *then* $\mathsf{n}(gs) = s^{-1}$.

By definition, $(\mathfrak{n}, \mathfrak{p})$ is a valid pair. The two following facts are obvious:

Proposition 45 *Let* (n, p) *be a valid pair for* (G, S). *Let* $\mathsf{next}(g) = g\mathsf{n}(g)$ *and* $\mathsf{prev}(g) = g\mathsf{p}(g)$. *Then* next *is one-to-one and onto with inverse* prev.

Proposition 46 *The set of all valid pairs for* (G, S) *is an SFT. More accurately, define:*

$$X_{G,\mathsf{S}} = \{x \in (\mathsf{S} \times \mathsf{S})^G | \forall s \in \mathsf{S}, \left[(x_g)_1 = s \Rightarrow (x_{gs})_2 = s^{-1}\right] \wedge \left[(x_g)_2 = s \Rightarrow (x_{gs})_1 = s^{-1}\right]\}$$

Then $X_{G,\mathsf{S}}$ *is an SFT. Furthermore, if* $x \in X_{G,\mathsf{S}}$, *then the pair* (n, p) *defined by* $\mathsf{n}(g) = (x_g)_1$ *and* $\mathsf{p}(g) = (x_g)_2$ *is a valid pair. Conversely, if* (n, p) *is a valid pair, then the configuration* x *defined by* $x_g = (\mathsf{n}(g), \mathsf{p}(g))$ *is in* $X_{G,\mathsf{S}}$.

Note that it might be possible that next and prev are not cyclic. In fact, in a typical configuration of $X_{G,\mathsf{S}}$, it is quite likely that $\mathsf{next}^i(g) = g$ for some g and $i > 0$.

Intuitively, a configuration x of $X_{G,\mathsf{S}}$ corresponds therefore to a partition of the Cayley graph $\Gamma(G, \mathsf{S})$ into cycles and bi-infinite paths, where, at each position $g \in G$, we should look at x_g to read the information coding $\mathsf{n}(g)$ and $\mathsf{p}(g)$ to know what is the next and previous vertex in the cycle or path. In group terms, we could say intuitively that we have partitioned the vertices of the Cayley graph of G into copies of the (canonical) Cayley graphs of $\mathbb{Z}$ and/or $\mathbb{Z}/n\mathbb{Z}$ for some (possibly infinitely many) n. We know also that, if S is chosen to verify the conclusion of Theorem 44, then at least one configuration of $X_{G,\mathsf{S}}$ contains only bi-infinite paths and no cycles.

We are now almost ready to proceed to the proof. Let G_1 and G_2 be two infinite, finitely generated groups and let $G = G_1 \times G_2$. Let S_1 and S_2 be two sets of generators for G_1 and G_2 that satisfy the conclusion of Theorem 44.

We then obtain two SFTs, X_{G_1,S_1} on G_1, and X_{G_2,S_2} on G_2. We extend X_{G_1,S_1} to an SFT on $G_1 \times G_2$ by extending periodically every configuration along G_2 (using the same idea as in Proposition 28), and proceed analogously with X_{G_2,S_2}. Taking the product of these two SFTs, we have now obtained an SFT X over the alphabet $\mathsf{S}_1 \times \mathsf{S}_1 \times \mathsf{S}_2 \times \mathsf{S}_2$ s.t.:

- If $(\mathsf{n}_1, \mathsf{p}_1)$ is a valid pair for (G_1, S_1) and $(\mathsf{n}_2, \mathsf{p}_2)$ a valid pair for (G_2, S_2) then the configuration x defined by $x_{(g_1,g_2)} = (\mathsf{n}_1(g_1), \mathsf{p}_1(g_1), \mathsf{n}_2(g_2), \mathsf{p}_2(g_2))$ is in X.
- Furthermore, every configuration of X is of this form.

Intuitively a configuration of X partitions the vertices of the Cayley graph $\Gamma(G, \mathsf{S}_1 \times \mathsf{S}_2)$ into copies of the (canonical) Cayley graphs of $\mathbb{Z} \times \mathbb{Z}$, $\mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}$, $\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$ and/or $\mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$. We know also that, as S_1 and S_2 were chosen to verify the conclusion of Theorem 44, that some configuration of X contains only copies of $\mathbb{Z} \times \mathbb{Z}$.

We are now ready for the proof of the undecidability of the domino problem on $G_1 \times G_2$. Let Y be a nearest neighbor SFT on $\mathbb{Z}^2$ with alphabet A. Let $\mathcal{F}_1$ be the set of forbidden patterns of Y of support $\{(0,0),(1,0)\}$ and $\mathcal{F}_2$ the set of patterns of support $\{(0,0),(0,1)\}$. We can interpret $\mathcal{F}_1$ and $\mathcal{F}_2$ as subsets of A^2, so that $y \in Y$ if and only if for all i, j $(y_{(i,j)}, y_{(i+1,j)}) \notin \mathcal{F}_1$ and $(y_{(i,j)}, y_{(i,j+1)}) \notin \mathcal{F}_2$

We can define a subshift Z on $G_1 \times G_2$ over the alphabet $(\mathsf{S}_1 \times \mathsf{S}_1 \times \mathsf{S}_2 \times \mathsf{S}_2 \times A)$ as follows:

$$Z = \left\{ z \in X \times A^{G_1 \times G_2} \middle| \forall g \in G_1 \times G_2 \begin{pmatrix} ((z_g)_5, (z_{g(z_g)_1})_5) \notin \mathcal{F}_1 \\ ((z_g)_5, (z_{g(z_g)_3})_5) \notin \mathcal{F}_2 \end{pmatrix} \right\}.$$

Basically, an element of $z \in Z$ is a configuration of X where each element g is additionally labeled with a letter from A. At each element g, we forbid the patterns of support $\{(0,0),(1,0)\}$ to appear in the direction indicated by $\mathsf{n}_1(g)$, and the patterns of support $\{(0,0),(0,1)\}$ to appear in the direction indicated by $\mathsf{n}_3(g)$.

Proposition 47 *Y is nonempty if and only if Z is nonempty.*

Proof: Suppose that Y is nonempty, and let y be an element of Y. Informally, we know some configuration of X partitions G into copies of $\mathbb{Z}^2$. We will thus use y in each copy of $\mathbb{Z} \times \mathbb{Z}$ to obtain our configuration of Z.

Formally, let $\mathfrak{next}_i, \mathfrak{prev}_i, \mathfrak{n}_i, \mathfrak{p}_i, h_i, i \in \{1, 2\}$ the functions that come from Theorem 44. Consider the configuration x of X that correspond to the valid pairs $(\mathfrak{n}_1, \mathfrak{p}_1, \mathfrak{n}_2, \mathfrak{n}_2)$.

We now define z by $(z_g)_{1-4} = (x_g)$ and $(z_{(g_1,g_2)})_5 = y_{\mathfrak{h}_1(g_1),\mathfrak{h}_2(g_2)}$. Then it is clear that $z \in Z$. Indeed, let $g = (g_1, g_2) \in G_1 \times G_2$. Then $(z_{g(z_g)_1})_5 = (z_{g(\mathfrak{n}_1(g_1),1_{G_2})})_5 = (z_{\mathfrak{next}_1(g_1),g_2})_5$ so that $(z_5, (z_{(z_g)_1 g})_5) = (y_{\mathfrak{h}_1(g_1),\mathfrak{h}_2(g_2)}, y_{\mathfrak{h}_1(g_1)+1,\mathfrak{h}_2(g_2)}) \notin \mathcal{F}_1$ by definition of y. Similarly, $(z_5, (z_{(z_g)_1 g})_5) \notin \mathcal{F}_2$. Therefore $z \in Z$ and Z is nonempty.

Conversely, suppose Z is nonempty and let $z \in Z$. Informally, z partitions G into copies of $\mathbb{Z} \times \mathbb{Z}$ or bastardized versions of it of the form $\mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}$. We look at just one of this copy, and use it to build our configuration on Y, possibly unfolding the copy if necessary.

Formally, let x be defined by $x_g = (z_g)_{1-4}$. Then $x \in X$, and therefore corresponds to two valid pairs $(\mathsf{n}_1, \mathsf{p}_2)$ and $(\mathsf{n}_2, \mathsf{p}_2)$. Let $\mathsf{next}_1, \mathsf{prev}_1$ and $\mathsf{next}_2, \mathsf{prev}_2$ be the associated functions. We now define $y \in A^{\mathbb{Z}^2}$ by $y_{(i,j)} = (z_{((\mathsf{next}_1)^i(1_G),(\mathsf{next}_2)^j(1_G))})_5$. Then $y \in Y$.

Indeed. Let $(i, j) \in \mathbb{Z}^2$. Let $g_1 = (\mathsf{next}_1)^i(1_G)$ and $g_2 = (\mathsf{next}_2)^j(1_G)$. Then

$$y_{(i+1,j)} = (z_{(\mathsf{next}_1)(g_1),g_2})_5 = (z_{(g_1\mathfrak{n}_1(g_1),g_2)})_5 = (z_{(g_1 z_{(g_1,g_2)})_1,g_2})_5$$

Therefore $(y_{(i,j)}, y_{(i+1,j)}) = ((z_{(g_1,g_2)})_5, (z_{((g_1 z_{(g_1,g_2)})_1,g_2)})_5) \notin \mathcal{F}_1$ by hypothesis on z. Similarly, $(y_{(i,j)}, y_{(i,j+1)}) \notin \mathcal{F}_2$. Therefore $y \in Y$ and Y is nonempty. ■

Corollary 48 *Let G_1, G_2 be two infinite, finitely generated groups. Then $G_1 \times G_2$ has an undecidable domino problem.*

Proof: Forbidden words for Z can be build effectively from a set of forbidden words of a nearest neighbor $\mathbb{Z}^2$-subshift. ■

Corollary 49 *The Grigorchuk group has an undecidable domino problem.*

Proof: The Grigorchuk group G is a well-known example of a torsion group. G contains a subgroup of finite index of the form $H = H_1 \times H_2$ with H_1, H_2 infinite [40]. $H_1 \times H_2$ has an undecidable domino problem by the previous propositions, therefore G has an undecidable domino problem by Proposition 29. ■

3.5 Discussion

3.5.1 Muller & Schupp theorem

In Section 3.4.1 we proved that virtually free groups have decidable domino problem. The proof directly gives an algorithm that decides the problem. It is noteworthy that this result can be obtained by a totally different argument. This comes from the combination of three facts: first, the domino problem can be expressed in Monadic Second Order logic (MSO) [58, 4]; second, a group is virtually free if and only if it has finite treewidth [41]; third, graphs with finite treewidth are exactly those with decidable MSO logic [33].

Proposition 50 ([41, 33]) *If G is virtually free, then G has decidable domino problem.*

The reasoning that leads to decidability of domino problem for virtually free groups using logics also suggests that this sufficient condition might also be necessary. This assumption comes from the following reasoning: if a group is not virtually free, then it has arbitrarily large grids as minors [47]. It should then be possible to somehow use these grids as computation zones – similarly to what is done in Robinson's tiling [48] – to encode Turing machine computations and conclude the undecidability of the domino problem. But the main issue is that even if we know that such grids exist, we do not know where they appear and even less how to make them appear inside a tiling by Wang tiles.

Conjecture 1 *A finitely presented group has decidable domino problem if and only if it is virtually free.*

3.5.2 Hyperbolic groups

The theorems in the previous section indicate that, whenever a group contains (in some sense) $\mathbb{Z} \times \mathbb{Z}$ or any other non-trivial Baumslag-Solitar group, they automatically have undecidable domino problem.

A specific class of groups where this result cannot be applied is the class of hyperbolic groups. Indeed, hyperbolic groups do not contain any copy of $\mathbb{Z} \times \mathbb{Z}$ or any other non-trivial Baumslag-Solitar group. They also always have decidable word problem, so Theorem 24 cannot apply, and they are always finitely presented.

Furthermore, free groups are hyperbolic, so it it very tempting to think that they always have decidable domino problem. However, this is not true.

Indeed, by a theorem of Rips [46], there exist hyperbolic groups G and finitely generated normal subgroups H of G s.t. G/H is isomorphic to $\mathbb{Z}^2$. Therefore, they have an undecidable domino problem by Proposition 28. It is also tempting to think the idea used above for the domino problem in the Baumslag-Solitar group can be extended for hyperbolic surface groups.

While these two ideas give examples of hyperbolic groups with undecidable domino problem, how to prove the result for an arbitrary, not virtually free, hyperbolic group, remains an open problem.

3.5.3 Translation-like and quasi-isometric groups

Section3.4.4 suggests that the undecidability of the domino problem is a geometric property: As soon as some Cayley graph of G contains a grid structure, G will have an undecidable domino problem. This is also hinted at in Section3.5.1. There are a few theorems that indeed suggest, at least for recursively presented groups, that it is indeed the case. In order to study this idea, we need the notion of quasi-isometry.

The definitions we give here are only valid in the context of finitely generated groups, but these notions can be defined in the general case of metric spaces.

Definition 26 *Let G_1, G_2 be two finitely generated groups and S_1 a set of generators for G_1.*

A map $f : G_1 \to G_2$ is Lipschitz if there exists a finite set $\mathsf{S}_2 \in H$ s.t. for all $g \in G_1$, and all $s \in \mathsf{S}_1$, $f(g)^{-1} f(gs) \in \mathsf{S}_2$.

A map $f : G_1 \to G_1$ is at bounded distance from the identity if there exists a finite set F for all $g \in G_1, g^{-1} f(g) \in F$

Compare the definition with Theorem 44. Geometrically, a Lipschitz map f sends adjacent vertices in the Cayley graph $\Gamma(G_1, \mathsf{S}_1)$ to adjacent (or identical) vertices in the Cayley graph $\Gamma(G_2, \mathsf{S}_2)$. It can be proven easily that the fact that f is Lipschitz does not depend on S_1, so that it is a property of the function and the group and not of the specific choice of generators.

Notice that if h is a homomorphism from G_1 to G_2, then h is a Lipschitz map by taking $\mathsf{S}_2 = h(\mathsf{S}_1)$.

Definition 27 *G_1 and G_2 are quasi-isometric if and only if there exists maps $f : G_1 \to G_2$, $g : G_2 \to G_1$ such that:*

- *f, g are Lipschitz.*
- *g is a quasi-inverse for f: $f \circ g$ and $g \circ f$ are at bounded distance from the identity.*

Quasi-isometry intuitively means that a pair of Cayley graphs of G_1 and G_2 look similar at large scale.

Definition 28 ([53, 59]) *A right action of G on H is an infix operator $\star : H \times G \to H$ s.t. $h \star 1_G = h$ and $h \star (g_1 g_2) = (h \star g_1) \star g_2$ for all $g_1, g_2 \in G$ and all $h \in H$.*

An action is free if $h \star g = h$ for some h implies $g = 1_G$

An action is translation-like if it is free and for all $g \in G$, the map $h \to h \star g$ is at bounded distance from the identity.

It is easy to see that it is sufficient to prove the last property for a set of generators of G. Therefore the last property can be replaced by: For some (any) set S_1 of generators of G, there exist S_2 that generate H s.t. for all $s \in \mathsf{S}_1$ and all $h \in H$, $h^{-1}(h \star s) \in \mathsf{S}_2$. Compare the definition with Theorem 44 when $G = \mathbb{Z}$.

An intuitive way to understand translation-like actions is that it covers some Cayley graph of H by copies of some Cayley graph of G.

Theorem 51 ([13]) *Let G and H be quasi-isometric finitely presented groups. G has undecidable domino problem if and only if H has undecidable domino problem.*

Theorem 52 ([25]) *Let G be a finitely presented group with undecidable domino problem and H a finitely generated group. If G acts translation-like on H, then H has an undecidable domino problem*

Both proofs are similar to the main proof of Section 3.4.4 in that they consist in seeing the Cayley graph of H as similar to the Cayley graph of G, or copies of the Cayley graph of G for the second theorem. The main difference is that we have replaced $\mathbb{Z}$, a free group, with an arbitrary *finitely presented* group. In the first theorem, they define a subshift of finite that codes all quasi-isometries that correspond to some finite sets $\mathsf{S}_1, \mathsf{S}_2$. In the second theorem, a subshift of finite type analogous to $X_{G,\mathsf{S}}$ from Corollary 48 will be defined that somehow codes all translation-like actions (and also non-free actions) that correspond to a function from a set of generators of G to a set of generators of H defined by the translation-like action. However, in this case the group G acting on H is not free, therefore the SFT needs also to code the relations of G (and hence needs G to be finitely presented for the construction to work).

4 Towards a definition of effective subshifts on groups

We start this section by presenting the notion of effectively closed subshift on $\mathbb{Z}^d$ for $d \geq 1$. This class of subshift appears naturally in dimension 1 as a generalization of sofic subshifts – those with a regular language, thus whose complement of the language is also regular – as follows: effectively closed subshifts are those which can be defined by a recursively enumerable set of forbidden patterns. In addition to this fact, effectively closed subshifts also appear in a natural way when studying some subsystems of two-dimensional SFTs or sofic subshifts, called projective subdynamics. These links are explained in Section 4.1.

4.1 Link between $\mathbb{Z}$ and $\mathbb{Z}^2$

To preserve the finiteness of the alphabet and thus compactness of the space of configurations, we use the projective subdynamics, to be defined in the Section 4.1.1. Note that this notion of projective subdynamics is different from the notion of subdynamics of a subshift defined by Hochman in [21].

4.1.1 Projective subdynamics: definition and example

We define the *projective subdynamics* of a two-dimensional subshift $X \subseteq A^{\mathbb{Z}^2}$ as the set of rows that can appear inside configurations of X :

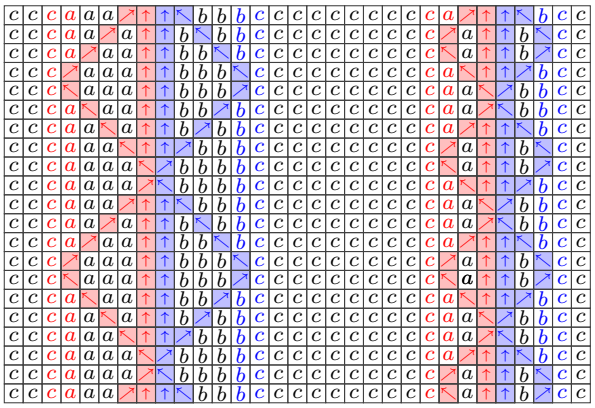
$$\pi(X) = \left\{ y \in A^{\mathbb{Z}} \mid \exists x \in X \text{ s.t. } x_{(i,0)} = y_i \text{ for every } i \in \mathbb{Z} \right\}.$$

The set of configurations $\pi(X)$ defined above is a subshift (see Exercise 2).

Example 53 *Consider the two dimensional SFT X on alphabet*

$$A = \left\{ a, a, b, b, c, c, c, \right.$$  ,  ,  ,  ,  ,  $\left.\right\}$

and whose allowed patterns appear in the following configuration.



The idea is to ensure that on every row, patterns of the form $a^n b^n$ appear in a sea of c' – without preoccupying of the colors. Diagonal signals are sent from the leftmost a' and the rightmost b' – they are designated as so if they have a c' as a neighbor. These diagonal signals bump on other signals and on symbols c' when they reach one, and two signals can collide only at the middle of a pattern $a^n b^n$ – the middle is marked with up-arrows.

It is left as an exercise to prove that the projective subdynamics of the SFT X is not a sofic subshift (see Exercise 3).

4.1.2 Effectively closed subshifts on $\mathbb{Z}^d$

A subshift $X \subseteq A^{\mathbb{Z}}$ is *effectively closed* if there exists a recursively enumerable set of forbidden patterns that defines it. To defined effectively closed subshifts on $\mathbb{Z}^d$ for $d \geq 2$, take any recursive bijection Φ between $\mathbb{Z}^d$ and $\mathbb{Z}$ – such a bijection always exists. Then a pattern p with finite support $S \subset \mathbb{Z}^d$ can be coded by the finite set of tuples $\Phi(p) = \{(\Phi(i), a) \mid i \in S \text{ and } p_x = a\}$. The set $\Phi(p)$ is called the pattern coding of p. We then define an *effectively closed* subshift on $\mathbb{Z}^d$ as a subshift for which there exists a recursively enumerable set of pattern codings that defines it.

From that definition, it is easy to check that the class of effectively closed subshifts contains SFTs and sofic subshifts.

Proposition 54 *Subshifts of finite type and sofic subshifts are effectively closed.*

Nevertheless, the class of effectively closed subshifts is wider than the class of sofic subshifts, as the following example in dimension 1 shows.

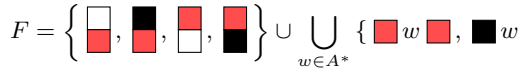
Example 55 *Let Y be the subshift defined as $\pi(X)$ where X is the two dimensional SFT of Example 53. This subshift Y is not sofic, nevertheless it can be defined as the set of configurations that avoid the following patterns*

$$\{ba, cb, ac\} \cup \{ca^m b^n c \mid m \neq n\}.$$

This set is obviously recursively enumerable, so that the subshift Y is effectively closed but not sofic.

The previous example cannot be adapted to dimension 2, since we lack an equivalent of graph representation of sofic subshifts in higher dimension. The difficulty lies in the fact that it is difficult to prove, in general, that a subshift is not sofic. There exist sufficient conditions for non-soficness, but no characterization exists. Nevertheless, there are explicit examples of effectively closed but non-sofic subshifts in dimension 2. The proof of non-soficness uses a combinatorial argument also used in [56], that can be generalized to prove non-soficness of well-chosen subshifts on amenable groups [1].

Example 56 *We define a two dimensional subshift X_{mirror}, called the* mirror shift*. It consists of all configurations over the alphabet $A = \{$ □, ■, ■ $\}$ which avoid the following patterns*

$F =$  ,

where $\tilde{w}$ denotes the mirror image of the word w, which is the word of length $|w|$ defined by $(\tilde{w})_i = w_{|w|-i+1}$ for all $1 \leq i \leq |w|$.

The mirror subshift X_{mirror} contains the $\mathbb{Z}^2$-fullshift $\{$ □, ■ $\}^{\mathbb{Z}^2}$ as a subsystem, but also all configurations that respect the following conditions: a symbol ■ *forces all symbols in the same column to be also* ■ *symbols; there is at most one column of* ■ *symbols; if a symbol* ■ *is present on a row, then* □ *and* ■ *symbols of this row are arranged symmetrically with respect to the* ■ *symbol.*

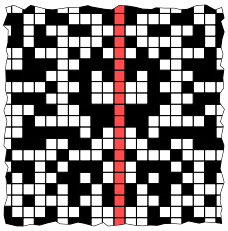


Figure 6: One configuration in the two-dimensional mirror subshift X_{mirror}.

The column of ■ *, if it appears in a configuration, behaves as a mirror towards the two half planes it defines, hence the name of the subshift. Obviously this subshift is effectively closed since the set of forbidden patterns F_{mirror} can be effectively enumerated, but one can prove it is not sofic by a direct combinatorial argument.*

Proposition 57 *The mirror subshift $X_{mirror} \subset A^{\mathbb{Z}^2}$ is not sofic.*

Proof: Consider $S = \{(0,1),(1,0)\}$ and suppose that the mirror subshift is sofic on $\mathbb{Z}^2$, then there exists a S-nearest neighbor $\mathbb{Z}^2$-SFT $X \subset B^{\mathbb{Z}^2}$ on some finite alphabet B and a 1-block factor code $\phi : X \twoheadrightarrow X_{\text{mirror}}$.

Let n be a positive integer and define $\Lambda_n := [-n,n]^2$. Notice that $\overset{\circ}{\Lambda}_{n+1} = \Lambda_n$ and thus $\partial\Lambda_{n+1} = \Lambda_{n+1} \setminus \Lambda_n$. In $L_{\Lambda_n}(X_{\text{mirror}})$ there are exactly $2^{(2n+1)^2}$ different patterns that do not contain a ■. These patterns are images of patterns of X with support $[-n,n]^2$ under ϕ and are surrounded with a crown with support $\partial\Lambda_{n+1}$. There are at most $|B|^{4(2n+2)}$ different crowns.

Consider now all configurations $x \in X_{\text{mirror}}$ in which a mirror appears at the origin, that is to say $x_{(0,j)} =$ ■ for all $j \in \mathbb{Z}$. For n large enough one has $|B|^{4(2n+2)} < 2^{(2n+1)^2}$, consequently there exist two distinct patterns P_1 and P_2 with support Λ_n that appear respectively in configurations y_1 and y_2 of X_{mirror} – assume that y_1 and y_2 are such that $(x_1)|_{\Lambda_n+(n^2,0)} = P_1$ and $(x_2)|_{\Lambda_n+(n^2,0)} = P_2$ – and such that there exist two distinct configurations x_1, x_2 in the extension X of X_{mirror} with the same crown – $(x_1)|_{\partial\Lambda_{n+1}+(n^2,0)} = (x_2)|_{\partial\Lambda_{n+1}+(n^2,0)}$ – and such that $y_1 = \phi(x_1)$ and $y_2 = \phi(x_2)$. As X is nearest neighbor we can construct a new configuration $\widetilde{y} \in A^{\mathbb{Z}^2}$ defined by

$$\widetilde{y}_z = \begin{cases} (P_2)_{z-(n^2,0)}, \text{ if } z \in \Lambda_n + (n^2,0) \\ (y_1)_z \text{ otherwise,} \end{cases}$$

in other terms $\widetilde{y}$ is the same configuration as y_1 except that pattern P_1 have been replaced by pattern P_2. On the one hand in configuration $\widetilde{y}$ a mirror appears at the origin, but since P_1 and P_2 have been chosen distinct $\widetilde{y} \notin X_{\text{mirror}}$. On the other hand the configuration $\widetilde{x} \in B^{\mathbb{Z}^2}$ defined by

$$\widetilde{x}_z = \begin{cases} (x_2)_{z-(n^2,0)}, \text{ if } z \in \Lambda_n + (n^2,0) \\ (x_1)_z \text{ otherwise,} \end{cases}$$

does not contain any forbidden pattern for X – that have been chosen nearest neighbor – and satisfies $\widetilde{y} = \phi(\widetilde{x})$, which proves that $\widetilde{y} \in X_{\text{mirror}}$ hence raising a contradiction.

■

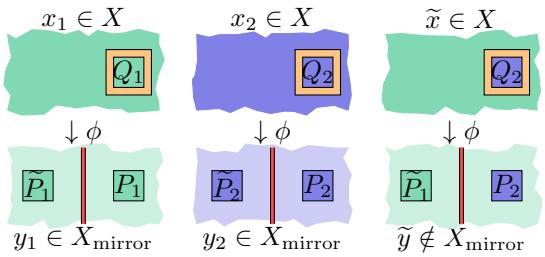


Figure 7: Two configurations y_1 and y_2 in the mirror subshift X_{mirror} with a mirror at the origin, and that differ on $\Lambda_n + (n^2, 0)$, but whose pre-images in the nearest neighbor $\mathbb{Z}^2$-SFT extension X are the same on $\partial\Lambda_{n+1}$. If it were so, one could construct a configuration $\widetilde{y}$ – by replacing $(y_1)|_{\Lambda_n+(n^2,0)}$ by $(y_2)|_{\Lambda_n+(n^2,0)}$ in configuration y_1 – which belongs to the image $\phi(X)$ but does not belong to X_{mirror}. This proves X_{mirror} is not sofic.

Remark 58 *One can define mirror subshifts in any dimension as the union of the $\mathbb{Z}^d$-fullshift $\{\square, \blacksquare\}^{\mathbb{Z}^d}$ and the set of configurations $x \in A^{\mathbb{Z}^d}$ with the hyperplane $\{i\} \times \mathbb{Z}^{d-1}$ filled with $\blacksquare$ symbols for some $i \in \mathbb{Z}$, and such that $x|_{\{i+j\}\times\mathbb{Z}^{d-1}} = x|_{\{i-j\}\times\mathbb{Z}^{d-1}}$ for every $j \in \mathbb{Z}$. Then Proposition 57 can be generalized to any dimension.*

We now list stability properties satisfied by effectively closed subshifts.

Proposition 59 *The class of effectively closed subshifts on $\mathbb{Z}^d$ is closed under finite intersection, finite union and morphism.*

Proof: The key ingredient is that one can choose a maximal – for inclusion – and recursively enumerable set of forbidden patterns to define an effectively closed subshift X. Indeed, the complement of the language of an effectively closed subshift has this property: it can be recursively enumerated from any recursively enumerable set of forbidden patterns that defines X, and is by definition maximal for inclusion.

If X_1 and X_2 are two effectively closed subshifts, then $X_1 \cap X_2$ (resp. $X_1 \cup X_2$) is defined by the set of forbidden patterns $\overline{\mathcal{L}_1} \cup \overline{\mathcal{L}_2}$ (resp. $\overline{\mathcal{L}_1} \cap \overline{\mathcal{L}_2}$). Since the union (resp. intersection) of two recursively enumerable languages is also recursively enumerable, effectively closed subshifts are closed under finite intersection (resp. finite union). Stability under morphisms comes from the fact that morphism are computable. ■

Proposition 60 *The class of effectively closed subshifts on $\mathbb{Z}^d$ is closed under projective subdynamics.*

Proof: This stability result follows from the fact that projective subdynamics are special cases of factors of subactions, and by Theorem 3.1 and Proposition 3.3 of [21] which establish that symbolic factors and subactions preserve effectiveness. ■

4.1.3 Simulation theorem

As seen in the previous section, neither the class of sofic subshifts nor the class of SFTs are not closed under projective subdynamics. Natural questions are thus: which subshifts can be obtained as projective subdynamics of sofic subshifts? of SFTs? For the first question, a complete characterization is known.

Theorem 61 (Hochman, [21]) *A subshift $Y \subseteq B^{\mathbb{Z}^k}$ is effectively closed if and only if it the the projective subdynamics of a sofic subshift $X \subseteq A^{\mathbb{Z}^{k+2}}$.*

The original construction by Hochman can be found in [21].

Theorem 62 (Aubrun & Sablik, [3], Durand, Romaschenko & Shen, [16]) *A subshift $Y \subseteq B^{\mathbb{Z}^k}$ is effectively closed if and only if it the the projective subdynamics of a sofic subshift $X \subseteq A^{\mathbb{Z}^{k+1}}$.*

These two theorems can be used as a blackbox to prove soficness of some complex subshifts, and lead to several applications. As a first example, consider the following result.

Theorem 63 (Myers, [42]) *There exist non-recursive two-dimensional SFTs, i.e. subshifts of finite type on some alphabet A such that none of their configurations can be described by a computable function $f : \mathbb{Z}^2 \to A$.*

This theorem was proven a few years after the publication of Robinson's proof of the undecidability of the domino problem. The proof is strongly inspired by Robinson's techniques, and is thus very technical. The same result can be obtained as a direct application of the simulation theorem as follow. Consider a one-dimensional effectively closed subshift X with no computable configurations (see [12] for an explicit construction). The two-dimensional sofic subshift, obtained by the simulation theorem, that projects onto X has no computable configurations (otherwise X would also contain a computable configurations). Since inverses of factor maps are computable, we get a non-recursive two-dimensional SFT.

Consider now S-adic subshifts, that are defined by a sequence of substitutions. In higher dimension and if the driving sequence is chosen to be computable, then S-adic subshifts are sofic [3]. This result generalizes theorems by Mozes [39] and Goodman-Strauss [17] on substitutive subshifts – when the driving sequence of substitutions is constant – to S-adic subshifts. The proofs of these two famous theorems are highly technical and difficult to handle. In contrast, the proof presented in [3] consists in a direct application of the simulation theorem, combined with a clever encoding of substitutions.

We list other examples of applications of the simulation theorem:

- In [50] the authors prove that tilings obtained by digitizing irrational vector spaces are aperiodic if and only if the digitized vector spaces are computable.
- In [26] it is proven that sets of periods for multidimensional SFTs are exactly sets of integers of the complexity class NP.
- In [5] an example of one-dimensional effectively closed subshift with a non-computable quasi-periodicity function is given.

4.2 Effectiveness on groups

This section is devoted to the study of what kind of subshifts on groups can be defined using Turing machines. As opposed to what happens in $\mathbb{Z}^d$ a general group might present extra difficulties which are related to its word problem.

Before defining the main object of this section, we make a digression with regards to their name. In the literature a subshift defined by a recursively enumerable set of forbidden patterns is usually said to be *effective*. Nevertheless, the name has also been used to talk about subshifts X whose language $L(X)$ is decidable, see for example [14]. These two notions define different objects. To avoid confusion we refer to these objects as *effectively closed subshifts*, as they would be called in computable analysis.

4.2.1 Definition and basic properties

For the definition of effectiveness in the context of subshift in a general group, we use Definition 22 of a pattern coding as the standard structure in which the information about forbidden patterns is given to a Turing machine.

Definition 29 *We say a subshift $X \subset A^G$ is* effectively closed *if there exists a recursively enumerable set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$.*

Usually in computability theory the word *effective* is used for objects defined by a decidable set instead of just a recursively enumerable one. In this case the usage of the word is justified, as effectively closed subshifts coincide with those which are definable by a decidable set of forbidden pattern codings.

Proposition 64 *Let $X \subset A^G$ be an effectively closed subshift. Then there exists a decidable set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$.*

Proof: Let $\mathcal{C}'$ a recursively enumerable set of pattern codings such that $X = X_{\mathcal{C}'}$. If $\mathcal{C}'$ is finite the result is trivial. Otherwise there exists a recursive enumeration $\mathcal{C}' = \{c_0, c_1, \dots\}$. For a pattern coding c we define its length as $|c| = \max_{(w,a)\in c} |w|$. For $n \in \mathbb{N}$ let $L_n = \max_{k\leq n} |c_k|$ and define $\mathcal{C}_n$ as the finite set of all pattern codings c which satisfy the following properties:

- Every $w \in S^*$ with $|w| \leq L_n$ appears in exactly one pair in c.
- $(w, a) \in c$ implies that $|w| \leq L_n$.
- If $(w, a) \in c_n$ then $(w, a) \in c$.

That is, $\mathcal{C}_n$ is the set of all pattern codings which are completions of c_n up to every word of length at most L_n in every possible way. Consider $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$. It is easy to check that

$$\bigcup_{c\in\mathcal{C}_n} \bigcap_{(w,a)\in c} [a]_w = \bigcap_{(u,b)\in c_n} [b]_u.$$

Therefore $X_{\mathcal{C}'} = X_{\mathcal{C}}$. We claim $\mathcal{C}$ is decidable.

Consider the algorithm which does the following on input c: It initializes n to 0. Then it enters into the following loop: First it produces the pattern coding c_n. If $L_n > |c|$ it rejects the input. Otherwise it calculates the set $\mathcal{C}_n$. If $c \in \mathcal{C}_n$ then it accepts, otherwise it increases the value of n by 1.

As L_n is increasing and cannot stay in the same value indefinitely this algorithm eventually ends for every input. ■

It is important to remark that the previous result just gives the existence of one decidable set of pattern codings which defines the subshift. Even in the case of $\mathbb{Z}$-subshifts there exist effectively closed subshifts whose language is undecidable. See Exercise 20.

Nevertheless, even with this result in hand, it is often more convenient to define effectively closed subshifts by a recursively enumerable set of forbidden pattern codings. In particular, if the group is recursively presented this set can be chosen canonically as the set of pattern codings which represent the complement of the language.

Proposition 65 *Let $X \subset A^G$ be an effectively closed subshift. If G is recursively presented then it is possible to choose $\mathcal{C}$ to be the recursively enumerable and maximal – for inclusion – set of pattern codings such that $X = X_{\mathcal{C}}$.*

Proof: A pattern coding c belongs to the maximal set $\mathcal{C}$ defining X if and only if $X \cap \bigcap_{(w,a)\in c}[a]_w = \emptyset$. Let $c \in \mathcal{C}$ and $\mathcal{C}'$ a recursively enumerable set such that $X = X_{\mathcal{C}'}$. Then:

$$\bigcap_{(w,a)\in c} [a]_w \subset \bigcup_{c'\in\mathcal{C}', g\in G} \bigcap_{(v,b)\in c'} [b]_{gv}.$$

By compactness we may extract a finite open cover indexed by c'_i, g_i such that:

$$\bigcap_{(w,a)\in c} [a]_w \subset \bigcup_{i\leq n} \bigcap_{(v,b)\in c'_i} [b]_{g_i v} \qquad (\star)$$

Note that each of these g_i can be seen as a finite word in S^*. Now let T be the Turing machine which does iteratively for $n \in \mathbb{N}$ the following:

- Runs n steps the machine T_1 recognizing $\mathtt{WP}(G)$ for every word in S^* of length smaller than n.
- Runs n steps the machine T_2 recognizing $\mathcal{C}'$ for every pattern coding defined on a subset of words of S^* of length smaller than n.
- Let $\sim_n$ be the equivalence relation for words in S^* of length smaller than n such that $u \sim_n v$ if uv^{-1} has been already accepted by T_1. Let $\mathcal{C}_n$ be the pattern codings already accepted by T_2. If every word in c has length smaller than n check if the following relation is true under $\sim_n$:

$$\bigcap_{(w,a)\in c} [a]_w \subset \bigcup_{c'\in\mathcal{C}_n, |u|\leq n} \bigcap_{(v,b)\in c'} [b]_{uv}$$

If it is true, accept, otherwise increase n by 1 and continue.

Let m be the max of all $|w|$ such that $(w,a) \in c$, and $|w'|$ such that $(w',a') \in c'_i$ and all $|g_i|$. By definition, there exists an $N \in \mathbb{N}$ such that every c'_i for $i \leq n$ is accepted and every word representing 1_G of length smaller than $2m$ is accepted. This means that at stage N relation ($\star$) is satisfied and T accepts c. If c is not in the maximal set, the machine never accepts. ■

In what follows, we show that the class of effectively closed subshifts is closed under factors and projective subdynamics when the group is recursively presented.

Proposition 66 *For recursively presented groups the class of effectively closed subshifts is closed under factors.*

Proof: Let $X \subset A_X^G$ be an effectively closed subshift. As G is recursively presented, a recursively enumerable set of pattern codings $\mathcal{C}_X$ defining X can be chosen to be maximal by Proposition 65. Consider a factor code $\sigma : X \twoheadrightarrow Y$ defined by a local function $\Phi : A_X^F \to A_Y$. Let $f_1, \dots, f_{|F|}$ be words in S^* representing the elements of F.

As σ is surjective, for each $a \in A_Y$ we have $|\Phi^{-1}(a)| > 0$. Therefore we can associate to a pair (w,a) a non-empty finite set of pattern codings

$$\mathcal{C}_{w,a} = \{(wf_i, p_{f_i})_{i=1,\dots,|F|} \mid p \in \Phi^{-1}(a)\}.$$

That is, $\mathcal{C}_{w,a}$ is a finite set of pattern codings over A_X representing every possible preimage of a. For a pattern coding $c = (w_i, a_i)_{i \leq n}$ where $a_i \in A_Y$ we define:

$$\mathcal{C}_c = \{ \bigcup_{(w,a)\in c} \widetilde{c}_{w,a} \mid \widetilde{c}_{w,a} \in \mathcal{C}_{w,a}\}.$$

That is, $\mathcal{C}_c$ is the finite set of pattern codings formed by choosing one possible preimage for each letter. Let $\mathcal{M}$ be the Turing machine which on entry c runs the machine recognizing $\mathcal{C}_X$ on every pattern coding in $\mathcal{C}_c$. If it accepts for every input, then $\mathcal{M}$ accepts c. Let $\mathcal{C}_Y$ be the set of pattern codings accepted by $\mathcal{M}$. We claim $Y = Y_{\mathcal{C}_Y}$.

Let $y \in Y_{\mathcal{C}_Y}$ and $n \in \mathbb{N}$. For each pattern coding c defined over all words of length at most n such that $y \in \bigcap_{(w,a)\in c}[a]_w$, there is a pattern coding $c_n \in \mathcal{C}_c$ which does not belong to $\mathcal{C}_X$. As $\mathcal{C}_X$ is maximal we have that $\bigcap_{(v,b)\in c_n}[b]_v \cap X \neq \emptyset$. Extracting a configuration x_n from $\bigcap_{(v,b)\in c_n}[b]_v \cap X$ we obtain a sequence $(x_n)_{n\in\mathbb{N}}$. By compactness there is a converging subsequence with limit $\widetilde{x} \in X$. By continuity of σ we have that $y = \sigma(\widetilde{x}) \in Y$. Conversely if $y \in Y$ there exists $x \in X$ such that $\sigma(x) = y$. Therefore for every finite $F' \subset G$ and pattern coding c such that $\bigcap_{(w,a)\in c}[a]_w = [y|_{F'}]$ there exists a pattern coding $\widetilde{c} \in \mathcal{C}_c$ such that $\bigcap_{(v,b)\in\widetilde{c}}[b]_v = [x|_{F'F}]$. Therefore, $c \notin \mathcal{C}_y$ and thus $y \in Y_{\,mathcalC_Y}$. ■

Definition 30 *Let $H \leq G$ be a subgroup of G. Given a subshift $X \subset A^G$ the H-projective subdynamics of X is the subshift $\pi_H(X) \subset A^H$ defined as:*

$$\pi_H(X) = \{x \in A^H \mid \exists y \in X, \forall h \in H, x_h = y_h\}$$

Proposition 67 *Let G be a recursively presented group and $H \leq G$ a finitely generated subgroup of G. If $X \subset A^G$ is effectively closed, then its H-projective subdynamics $\pi_H(X)$ is effectively closed.*

Proof: As H is finitely generated, there exists a finite set $\mathsf{S}' \subset H$ such that $\langle \mathsf{S}' \rangle = H$. As G is finitely generated by S there exists a function $\gamma : \mathsf{S}' \to \mathsf{S}^*$ such that $s' =_G \gamma(s')$ (that is, every element of S' can be written as a word in S^*). Extend the function γ to act by concatenation over words in S'^*.

As G is recursively presented, by Proposition 65 the set of pattern codings $\mathcal{C}_G$ defining X can be chosen to be maximal. Let $c = (w_i, a_i)_{i\in I}$ a pattern coding where $w_i \in \mathsf{S}'^*$ and define $\gamma(c) := (\gamma(w_i), a_i)_{i\in I}$. Let $\mathcal{M}$ be the Turing machine which on entry c runs the algorithm recognizing $\mathcal{C}_G$ on entry $\gamma(c)$ and accepts if and only if this machine accepts. Clearly $\mathcal{C}_H = \{c \mid \mathcal{M} \text{ accepts } c\}$ is recursively enumerable. Also, as $\mathcal{C}_G$ is a maximal set of pattern codings then $c \in \mathcal{C}_H \iff \bigcap_{(w,a)\in\gamma(c)}[a]_w \cap X = \emptyset$. Therefore $\pi_H(X) = X_{\mathcal{C}_H}$. ■

4.2.2 The case of non recursively presented groups

In order to prove some of the properties of effectively closed subshifts we have used the hypothesis that the group is recursively presented. What could go wrong if this is not the case? We aim to throw some light into this question.

The main problem we encounter is the failure of Proposition 65. Indeed, it even fails for the simplest example. The full shift is always effectively closed as the empty set of pattern codings defines it. Nevertheless if the alphabet A contains at least two symbols then a maximal set of forbidden pattern codings contains in particular the coding $\{(\epsilon, a), (w, b)\}$ for $b \neq a$ if and only if $w \in \mathtt{WP}(G)$. If this set were recursively enumerable one could use it to enumerate $\mathtt{WP}(G)$ and thus G would be recursively presented.

Consider the case of a sofic subshift Y in a non recursively presented group G. Any SFT extension $\sigma : X \twoheadrightarrow Y$ can be represented by a finite set of pattern codings for X and some fixed coding of the finite set F which defines the block code $\Phi : A_X^F \to A_Y$ which determines σ. A recursively enumerable set of pattern codings for Y would consist of a list of patterns –which in particular– do not appear in Y. This means that every pattern obtained by taking the preimage under Φ does not appear in X. Nevertheless, the previous argument implies that we can not test this algorithmically in general. This is the reason why the proof of Proposition 66 does not extend to this case.

4.2.3 The one-or-less subshift $X_{\leq 1}$

Consider the subshift $X_{\leq 1} \subset \{0,1\}^G$ whose configurations contain at most one appearance of the letter 1.

$$X_{\leq 1} = \{x \in \{0,1\}^G \mid 1 \in \{x_g, x_h\} \Rightarrow g = h\}$$

We are going to show that even in the case of recursively presented groups this subshift can fail to be effectively closed.

Proposition 68 *If G is infinite, then $X_{\leq 1}$ is not an SFT.*

Proof: Suppose $X_{\leq 1} = X_{\mathcal{F}}$ for a finite $\mathcal{F}$ and let $F = \bigcup_{p \in \mathcal{F}} \mathrm{supp}(p)$, $U = \bigcup_{h \in F^{-1}} hF$ and note that $|U| < \infty$. As G is infinite, there exists $g \in G \setminus U$. Consider the configuration $x \in \{0,1\}^G$ which takes the value 1 in $\{1_G, g\}$ and 0 elsewhere. Clearly $x \notin [p]_h$ for every $h \in G$ and $p \in \mathcal{F}$ otherwise $\{1_G, g\} \subset hF$ implying that $hF \subset U$ and thus $g \in U$. Therefore $x \in X_{\mathcal{F}}$ but $x \notin X_{\leq 1}$. ■

Proposition 69 *Let G be a recursively presented group. Then $X_{\leq 1}$ if effectively closed if and only if the word problem of G is decidable.*

Proof: If $\mathtt{WP}(G)$ is decidable then $X_{\leq 1}$ is effectively closed. Indeed, an algorithm recognizing a maximal set of pattern codings $\mathcal{C}$ such that $X_{\leq 1} = X_{\mathcal{C}}$ is the following: On input c it considers every pair $(w_1, 1), (w_2, 1)$ in c and accept if and only if $w_1 w_2^{-1} \neq_G 1_G$ for a pair. Conversely, as G is recursively presented, the word problem is already recursively enumerable. It suffices to show it is co-recursively enumerable.

By Proposition 65 there exists a maximal set of forbidden pattern codings $\mathcal{C}$ with $X_{\leq 1} = X_{\mathcal{C}}$. Given $w \in S^*$, consider the pattern coding $c_w = \{(\epsilon, 1), (w, 1)\}$. Note that $w \neq_G 1_G \iff c_w \in \mathcal{C}$. Therefore the the algorithm which on entry $w \in S^*$ runs the algorithm recognizing $\mathcal{C}$ on entry c_w and accepts if and only if this one accepts, recognizes $S^* \setminus \mathtt{WP}(G)$. Hence $\mathtt{WP}(G)$ is co-recursively enumerable. ■

Using Proposition 66 we obtain the following corollary.

Corollary 70 *If G is recursively presented and $\mathtt{WP}(G)$ is undecidable, then $X_{\leq 1}$ is not a sofic subshift.*

4.3 Two larger notions of effectiveness

As stated in Section 4.2, the classical notion of effectively closed $\mathbb{Z}^d$-subshifts extended to finitely generated groups fails to be completely satisfying for several reasons. First, the notion of effectively closed subshift is not directly related to the group G itself: we use pattern codings, which is in some sense a way to come back to dimension 1. Another way to formulate this reservation it that, unlike the case of $\mathbb{Z}$ where effectively closeness comes with a natural computational model –classical Turing machines– there is no similar correspondence for effectively closed subshifts on a finitely generated group G. And second, very simple subshifts like the one-or-less subshift $X_{\leq 1}$ are not effectively closed for recursively presented groups with undecidable word problem (Proposition 69): it would be natural to ask that this subshift is always effective, independently of the complexity of the word problem of the group.

In order to escape from these limitations, we study two different extensions of the notion of effectiveness which cover a larger countable class of subshifts. These notions are G-effectiveness and enumeration-effectiveness.

The notion of G-effectiveness tackles the problems linked to the word problem of the group by adding the language $\mathtt{WP}(G)$ as an oracle. The advantage of this class, besides repairing the previous problems related to the word problem of the group, is that the set of subshifts defined by it can be given a natural definition using modified Turing machines which use the group as a tape. This characterization is interesting in the sense that it allows explicit constructions to be made with the help of these machines. Some examples can be found in [1].

Then we show the limitations of the notion of G-effectiveness, and propose an alternative definition of effectiveness for G-subshifts called enumeration effectiveness. This new notion, which is weaker than G-effectiveness, allows us to generalize Proposition 69 to any group – the recursive presentation hypothesis is no longer needed.

4.3.1 G-effectiveness

Definition 31 *A subshift $X \subset A^G$ is G-effectively closed if there is a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and $\mathcal{C}$ is recursively enumerable with oracle $\mathtt{WP}(G)$.*

Following from the results of this section, one can directly use the definition above to show that the following properties hold for any finitely generated group G.

1. If X a G-effectively closed subshift then a maximal set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$ is recursively enumerable with oracle $\mathtt{WP}(G)$.
2. The class of G-effectively closed subshifts is closed under finite intersections and unions.
3. The class of G-effectively closed subshifts is closed under factors.
4. Being G-effectively closed is a conjugacy invariant.
5. The class of G-effectively closed subshifts contains all sofic subshifts.
6. The class of G-effectively closed subshifts contains all effectively closed subshifts.
7. If $\mathtt{WP}(G)$ is decidable, then every G-effectively closed subshift is effectively closed.
8. $X_{\leq 1}$ is a G-effectively closed subshift.

Nevertheless, this class fails to be stable under projective subdynamics.

Proposition 71 *Let G be a group which is not recursively presented. There exists a $(G \times \mathbb{Z})$-effectively closed subshift $X \subset A^{G \times \mathbb{Z}}$ such that its $\mathbb{Z}$-projective subdynamics is not $\mathbb{Z}$-effectively closed.*

Proof: Let $A = S \cup \{\star\}$. For $w \in S^*$, let p_w defined over the support $\{1_G\} \times \{0, \dots |w|+1\}$ such that $(p_w)_{(1_G,0)} = (p_w)_{(1_G,|w|+1)} = \star$ and for $j \in \{1, \dots, |w|\}$ then $(p_w)_{(1_G,j)} = w_j$. Let $X := X_{\mathcal{F}} \subset A^{G \times \mathbb{Z}}$ be defined by the set of forbidden patterns $\mathcal{F} = \{p_w \mid w \in \mathtt{WP}(G)\}$. Clearly X is $(G \times \mathbb{Z})$-effectively closed. Every $\mathbb{Z}$-coset of a configuration $x \in X$ contains a bi-infinite sequence $y \in A^{\mathbb{Z}}$ such that either y contains at most one symbol $\star$ or every word appearing between two appearances of $\star$ represents 1_G in G.

We claim that $\pi_{\mathbb{Z}}(X)$ is not effectively closed. If it were, there would exist a maximal set of forbidden pattern codings which is recursively enumerable and defines $\pi_{\mathbb{Z}}(X)$. Therefore

given $w \in S^*$ a machine could run the algorithm for the word $\star w \star$ and it would be accepted if and only if $w =_G 1_G$. This would imply that G is recursively presented. ■

Although Proposition 71 is a theoretical drawback for the notion of G-effectiveness, it is noteworthy that this behavior only happens when the projective subdynamics is taken with respect to a group with strictly weaker word problem. If the projective subdynamics of a $(G \times \mathbb{Z})$-effectively closed subshift is taken with respect to G, the resulting subshift is indeed G-effectively closed.

The interest of this class is mainly due to the fact that they can be defined in a natural way using modified Turing machines. These objects which we call *G-machines* replace the bi-infinite tape by a Cayley graph $\Gamma(G, \mathsf{S})$ of the finitely generated group G.

Definition 32 *A G-machine is a 6-tuple $(Q, \Sigma, \sqcup, q_0, Q_F, \delta)$ where Q is a finite set of states, Σ is a finite alphabet, $\sqcup \in \Sigma$ is the blank symbol, $q_0 \in Q$ is the initial state, $Q_F \subset Q$ is the set of accepting states and $\delta : \Sigma \times Q \to \Sigma \times Q \times \mathsf{S}$ is the transition function.*

G-machine T acts on the set $\Sigma^G \times Q$ as follows: let $(x, q) \in \Sigma^G \times Q$ and $\delta(x_{1_G}, q) = (a, q', s)$. Then $T(x, q) = (S_{s^{-1}}(\widetilde{x}), q')$ where $\widetilde{x}|_{1_G} = a$ and $\widetilde{x}|_{G \setminus \{1_G\}} = x|_{G \setminus \{1_G\}}$. Figure 8 illustrates this action when G is a free group. Here the head of the Turing machine is assumed to stay at a fixed position and the tape moves instead.

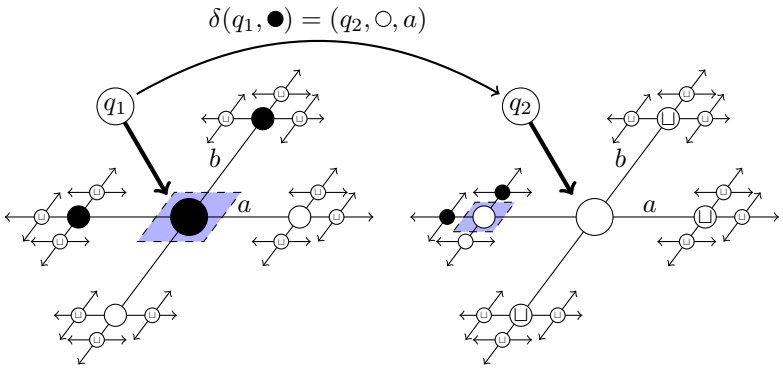


Figure 8: A transition of an G-machine for the free group on two elements.

Let $F \subset G$ be a finite set and $p \in \Sigma^F$. Let $x^p \in \Sigma^G$ be the configuration such that $(x^p)|_F = p$ and $(x^p)|_{G \setminus F} \equiv \sqcup$. We say that T *accepts* p if there is $n \in \mathbb{N}$ such that $T^n(x^p, q_0) \in \Sigma^G \times Q_F$. $L \subset L(A^G)$ is *G-recursively enumerable* if there exists a G-machine T which accepts $p \in \Sigma_G^*$ if and only if $p \in L$. If both L and $L(A^G) \setminus L$ are G-recursively enumerable we say L is *G-decidable*.

In [1] it is shown that these machines characterize G-effective subshifts. We say a set of patterns $L \subset L(A^G)$ is *closed by extensions* if for each $p_1, p_2 \in L(A^G)$ such that $p_1 \sqsubset p_2$ then $p_1 \in L \implies p_2 \in L$. Also, for a set of pattern codings $\mathcal{C}$ we denote by $\mathsf{p}(\mathcal{C})$ the set of patterns they define in the group G. Formally:

$$\mathsf{p}(\mathcal{C}) = \{p \in L(A^G) \mid \exists c \in \mathcal{C}, [p] = \bigcap_{(w,a) \in c} [a]_w\}$$

Theorem 72 *Let G be a finitely generated infinite group and $L \subset L(A^G)$ be a set of patterns. If L is G-recursively enumerable then there exists a recursively enumerable with oracle $\mathtt{WP}(G)$ set of pattern codings $\mathcal{C}$ such that $L = \mathsf{p}(\mathcal{C})$. Conversely, if $\mathcal{C}$ a recursively enumerable with oracle $\mathtt{WP}(G)$ set of pattern codings. If $\mathsf{p}(\mathcal{C})$ is closed by extensions, then $\mathsf{p}(\mathcal{C})$ is G-recursively enumerable.*

Using the fact that the maximal set of forbidden patterns of a subshift is closed by extension, we obtain what follows.

Corollary 73 *A subshift $X \subset A^G$ is G-effectively closed if and only if there exists a G-recursively enumerable set $\mathcal{F} \subset L(A^F)$ such that $X = X_{\mathcal{F}}$.*

A big drawback of the class of G-effectively closed subshifts is that in general they do not admit a simulation theorem in the sense of Theorem 62. Specifically, $X_{\leq 1}$ is G-effective for each group, but it cannot be obtained as the projective subdynamics of a sofic subshift if G is finitely generated, recursively presented group but has undecidable word problem.

Proposition 74 *Let G be a finitely generated, recursively presented group with undecidable word problem and H a finitely generated group such that $\mathtt{WP}(H) \leq_m \mathtt{WP}(G)$. Then $X_{\leq 1}$ cannot be obtained as the G-projective subdynamics of a sofic $G \times H$-subshift.*

Proof: As G is recursively presented and $\mathtt{WP}(H) \leq_m \mathtt{WP}(G)$, then H is also recursively presented and thus $G \times H$ is recursively presented. Applying Proposition 66 we get that sofic $G \times H$-subshifts are effectively closed. Hence, using Proposition 67 the G-projective subdynamics must also be effectively closed. We conclude as per the fact that $X_{\leq 1}$ is not effectively closed for recursively presented groups with undecidable word problem.

In particular, this means that if G is a finitely generated, recursively presented group with undecidable word problem, then there is no general simulation theorem for G-effective subshifts coming from sofic subshifts on $G \times \mathbb{Z}^d$ or even $G \times \cdots \times G$. It might therefore be interesting to weaken the notion of G-effectiveness so that even if $X_{\leq 1}$ is no longer always in the class, a general simulation theorem is not proscribed.

4.3.2 Enumeration effectiveness

If A and B are two sets of words in S, we say that A is *enumeration reducible* to B, denoted $A \leq_e B$, if there exists an algorithm that produces an enumeration of A from any enumeration of B. Formally, $A \leq_e B$ if there exists a partial computable function f that associates to each $\langle n, i \rangle$ a finite set $D_{n,i}$ such that $n \in A \iff \exists i, D_{n,i} \subseteq B$.

Remark 75 *A set A is recursively enumerable if and only if $A \leq_e \emptyset$. Particularly, if A is recursively enumerable then $A \leq_e B$ for any set B.*

From this remark, we get a characterization of effectively closed subshifts as those for which there exists a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and $\mathcal{C} \leq_e \emptyset$.

We will use a characterization of enumeration reducibility, that can be found in [44, Exercise XIV.1.2] and in [52].

Proposition 76 *$A \leq_e B$ if and only if for every set C, if B is recursively enumerable with oracle C then A is also recursively enumerable with oracle C.*

We can translate the notion of G-effectiveness in term of enumeration reducibility. If we denote $B \oplus C$ the set $\{(0, x) \mid x \in B\} \cup \{(1, x) \mid x \in C\}$, we get that A is recursively enumerable with oracle B if and only if $A \leq_e B \oplus \overline{B}$. As written in Definition 31, a subshift $X \subset A^G$ is G-effectively closed if there is a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and $\mathcal{C}$ is recursively enumerable with oracle $\mathtt{WP}(G)$. We thus get the following characterization of G-effectively closed subshifts.

Proposition 77 *A G-subshift $X \subset A^G$ is G-effectively closed if there is a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and such that $\mathcal{C} \leq_e \mathtt{WP}(G) \oplus \overline{\mathtt{WP}(G)}$.*

Thus with G-effectiveness, forbidden patterns are produced from two enumerations: one enumeration of the words in S^* that represent the identity of the group (the word problem $\mathtt{WP}(G)$), and one enumeration of the word in S^* that do not represent the identity of the group (the complement of the word problem $\overline{\mathtt{WP}(G)}$). But are those two enumerations strictly necessary to produce forbidden patterns? In order to check whether a pattern coding is inconsistent, it suffices to check all pairs of words w, w', and if at some point two words happen to represent the same group element, check whether they are assigned different symbols. Only the enumeration of the word problem is needed for that. The notion of enumeration effectiveness is based on this observation.

Definition 33 *A G-subshift $X \subset A^G$ is* G-enumeration effective *if there exists a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$ and $\mathcal{C} \leq_e \mathtt{WP}(G)$.*

We first compare the class of G-enumeration effective subshifts with the classes of effectively closed subshifts and G-effectively closed subshifts. From the characterizations of these classes with enumeration reduction, it follows immediately that G-enumeration effective subshifts are in-between the two others. The diagram on Figure 9 summarizes the propositions listed below.

Proposition 78 *Let G be a finitely generated group and X an effectively closed subshift. Then X is G-enumeration effective.*

Proof: Since X is G-effectively closed, there exists a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and $\mathcal{C} \leq_e \emptyset$. A fortiori, we get that $\mathcal{C} \leq_e \mathtt{WP}(G)$, thus X is G-enumeration effective. ■

Proposition 79 *Let G be a finitely generated group and X a G-enumeration effective subshift. Then X is G-effectively closed.*

Proof: Since X is G-enumeration effective, there exists a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and $\mathcal{C} \leq_e \mathtt{WP}(G)$. A fortiori, we get that $\mathcal{C} \leq_e \mathtt{WP}(G) \oplus \overline{\mathtt{WP}(G)}$, thus X is G-effectively closed. ■

Conversely, if $\overline{\mathtt{WP}(G)}$ is enumeration reducible to $\mathtt{WP}(G)$, any set enumeration reducible to $\mathtt{WP}(G)$ is also enumeration reducible to $\mathtt{WP}(G) \oplus \overline{\mathtt{WP}(G)}$, from which we deduce that groups with the property are exactly groups where G-enumeration effectiveness and G-effectiveness coincide.

Proposition 80 *Let G be a finitely generated such that $\overline{\mathtt{WP}(G)} \leq_e \mathtt{WP}(G)$ and X a G-effectively closed subshift. Then X is G-enumeration effective.*

Proposition 81 *G-enumeration effectiveness is closed under factors. In particular, if X is a sofic subshift, then it is G-enumeration effective.*

Proof: Recall that a finitely generated group is recursively presented if and only if $\mathtt{WP}(G)$ is recursively enumerable. Let $\mathcal{C}$ be the list of forbidden pattern codings obtained in the proof of Proposition 66 (with an oracle to $\mathtt{WP}(G)$ in this case). We have shown that $\mathcal{C}$ is recursively enumerable if $\mathtt{WP}(G)$ is recursively enumerable.

Let C be an arbitrary language such that $\mathtt{WP}(G)$ is recursively enumerable with oracle C, then obviously $\mathcal{C}$ is recursively enumerable with oracle C. By Proposition 76 this implies that $\mathcal{C} \leq_e \mathtt{WP}(G)$. This shows that G-enumeration effectiveness is closed under factors.

Let Y be an SFT extension of X. Clearly Y is always effectively closed: Take a finite list of forbidden patterns defining it hard code each pattern into a pattern coding. By Proposition 78 we have that Y is G-enumeration effective. As G-enumeration effectiveness is closed under factors, we conclude that X is also G-enumeration effective. ■

Proposition 82 *Let G be a finitely generated group. Then the one-or-less subshift $X_{\leq 1}$ is G-enumeration effective if and only if $\overline{\mathtt{WP}(G)} \leq_e \mathtt{WP}(G)$.*

This is not a vacuous hypothesis: if G is recursively presented (hence $\mathtt{WP}(G)$ is recursively enumerable), this implies that $\overline{\mathtt{WP}(G)}$ is also recursively enumerable, hence recursive. Contrary to Proposition 69, this characterization of the effectiveness of the one-or-less subshift does not require the group G to be recursively presented.

Proof: As claimed in Section 4.3.2, the subshift $X_{\leq 1}$ is G-effectively closed, which means that there is a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and such that $\mathcal{C} \leq_e \mathtt{WP}(G) \oplus \overline{\mathtt{WP}(G)}$ by Proposition 77. Suppose that $\overline{\mathtt{WP}(G)} \leq_e \mathtt{WP}(G)$. Let C be a set such that $\mathtt{WP}(G)$ is recursively enumerable with oracle C, and denote $\mathcal{M}$ the Turing machine with oracle C that recognizes $\mathtt{WP}(G)$. We construct a new Turing machine $\mathcal{M}'$ with oracle C with the following behavior. Given a pattern coding c, $\mathcal{M}'$ in parallel simulates $\mathcal{M}_e$ to enumerate the set $\{0\} \times \mathtt{WP}(G)$ and recursively enumerates $\{1\} \times \overline{\mathtt{WP}(G)}$ from the latter enumeration. From this enumeration of $\mathtt{WP}(G) \oplus \overline{\mathtt{WP}(G)}$, the machine $\mathcal{M}'$ then produces an enumeration of $\mathcal{C}$. Thus $\mathcal{C} \leq_e \mathtt{WP}(G)$.

Reciprocally, first note that in an analogous way to Proposition 65 we can suppose that any G-enumeration effective subshift is given by a maximal set of forbidden pattern codings. Let $X_{\leq 1}$ be G-enumeration effective and $\mathcal{C} \leq_e \mathtt{WP}(G)$ be the maximal set of pattern codings

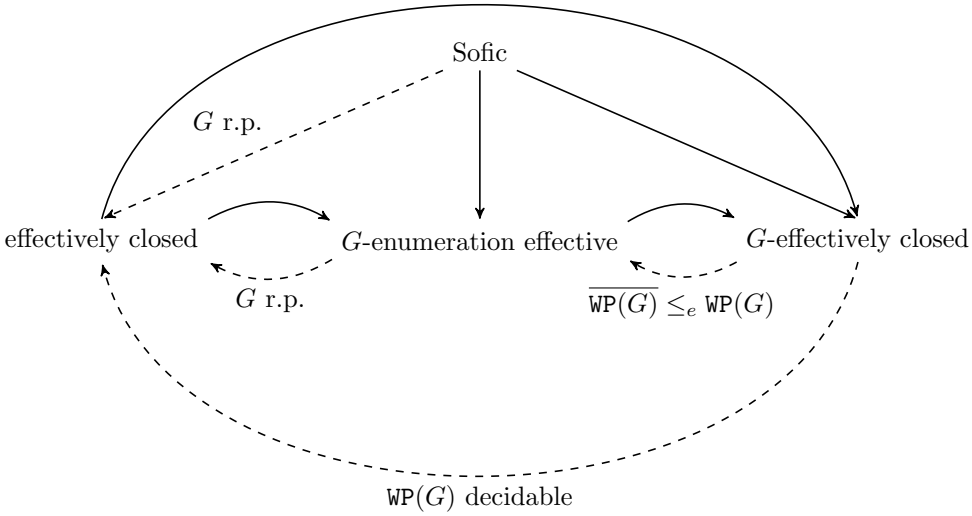


Figure 9: Inclusion relations between different classes of G-subshifts for a finitely generated group G. Inclusion represented by a dashed arrow only holds for groups having the property labelling the arrow.

defining $X_{\leq 1}$. As in the proof of Proposition 69, we can define $f : S^* \to \mathcal{C} \cup \overline{\mathcal{C}}$ where $f(w) = \{(\epsilon, 1), (w, 1)\}$. Clearly if $w \in \overline{\mathtt{WP}(G)}$ if and only if $f(w) \in \mathcal{C}$. Therefore $\overline{\mathtt{WP}(G)} \leq_m \mathcal{C} \leq_e \mathtt{WP}(G)$ which implies $\overline{\mathtt{WP}(G)} \leq_e \mathtt{WP}(G)$. ■

4.3.3 Towards a simulation theorem

In the case of $\mathbb{Z}^d$-subshifts, the notion of effectively closed subshift is quite natural for two reasons. First it extends the notion of sofic $\mathbb{Z}$-subshift from the point of view of pattern exclusion: a sofic $\mathbb{Z}$-subshift has a regular language, while an effectively closed $\mathbb{Z}$-subshift has a recursively enumerable language. Hence sofic subshifts are effectively closed. Second the class of effectively closed subshifts is stable under projective subdynamics (see Section 4.1). A reasonable generalization of effectiveness to finitely generated groups should at least satisfy these two properties, with the hope that a simulation theorem may hold.

Clearly, the notion of effectiveness for a finitely generated group G given in Definition 29 is too restrictive: sofic subshifts are effectively closed for recursively presented groups (Proposition 66), but we do not know what happens for non recursively presented groups. Moreover, this notion does not behave well with projective subdynamics (see Propositions 71). By Propositions 81 and 79, sofic subshifts are always G-enumeration effective, and thus G-effectively closed. Thus among the three definitions presented in this chapter, only G-effectiveness and G-enumeration effectiveness are likely to fulfill our requirements. Another argument that may be taken into account to choose between these two notion is the one-or-less subshift $X_{\leq 1}$. On the one hand, this subshift is always G-effectively closed (Section 3.4.1), on the other hand, it is G-enumeration effective only for finitely generated groups satisfying that $\overline{\mathtt{WP}(G)} \leq_e \mathtt{WP}(G)$.

What would be a general statement of a simulation theorem for a finitely generated group G? The two notions of G-enumeration effectiveness and G-effectiveness both depend on the group G. Apart from torsion groups and with no additional restriction, the operation of projective subdynamics may transform a G-subshift on a $\mathbb{Z}$-subshift, where the three notions of effectiveness coincide. So we should consider only projective subdynamics from a group to a subgroup with the same complexity of $\mathtt{WP}$. The simulation theorems from [21, 6] suggest that adding the group $\mathbb{Z}^2$ to the group G – what is meant by *adding* has of course to be precise – makes possible a characterization of projective subdynamics of sofic subshifts on G equipped

with $\mathbb{Z}^2$ as effective G-subshifts. In Section 3.4.4, the result of [53] is used to extract a grid structure from a direct product $G_1 \times G_2$ where G_1 and G_2 are both infinite. Hence the idea to study projective subdynamics of $G \times G \times G$-effectively closed subshifts. Unfortunately, Proposition 74 tells us that for recursively presented groups G with undecidable $\mathtt{WP}$, some simple G-effectively closed subshifts cannot be realized as the projective subdynamics of a sofic $G \times H$-subshift, where H is any finitely generated group with $\mathtt{WP}(H) \leq_m \mathtt{WP}(G)$. So a simulation theorem is excluded for G-effectiveness. To the knowledge of the authors, the question remains open for G-enumeration effectiveness.

5 Exercises

Exercise 1 *Show that if $L \leq_m L'$, then $L \leq_T L'$ **Hint**: Express $L \leq_m L'$ with a Turing machine with oracle L'.*

Exercise 2 *Let $X \subseteq A^{\mathbb{Z}^2}$ be a subshift. Show that*

$$\pi(X) = \left\{ y \in A^{\mathbb{Z}} \mid \exists x \in X \text{ s.t. } x_{(i,0)} = y_i \text{ for every } i \in \mathbb{Z} \right\}$$

is also a subshift.

Exercise 3 *Prove that the projective subdynamics of the subshift defined in Example 53 is not sofic. **Hint**: Use the fact that the language $\{a^n b^n \mid n \in \mathbb{N}\}$ is not a regular language.*

Exercise 4 *Give an example of a recursively presented group which is not finitely presented.*

Exercise 5 *Show that the properties of being an SFT and being a sofic subshift are conjugacy invariants.*

Exercise 6 *Show that the domino problem is a group isomorphism invariant, that is, that if G is isomorphic to H then the domino problem of G is many-one equivalent to the domino problem of H.*

Exercise 7 *Find an example of a finitely generated group which contains a non-finitely generated subgroup. **Hint**: One possibility is to use Corollary 33.*

Exercise 8 *Let G be a finitely generated group and $H \trianglelefteq G$. Show that the quotient group G/H is also finitely generated.*

Exercise 9 *Let $H \leq G$ be a subgroup such that $[G : H] < \infty$. Show that there exists $N \leq H$ such that $N \trianglelefteq G$ and $[G : N] < \infty$. **Hint**: define N as the stabilizer of the action of G over the lateral classes G/H by left multiplication.*

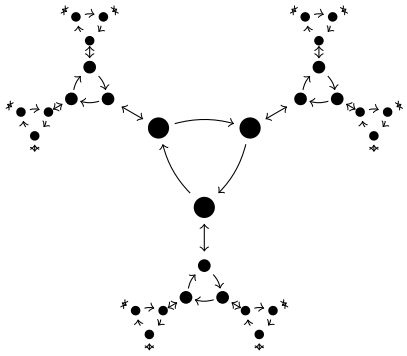
Exercise 10 *Let G be a finitely generated group and $H \leq G$ a subgroup of finite index. Show that H is finitely generated. **Hint**: Let $L = \{\ell_1 \dots, \ell_n\}$ be a set of representatives of the left lateral classes containing 1_G. For each generator s of G write $s\ell_i = \ell_{i,s} h_{i,s}$ for some $\ell_{i,s} \in L$ and $h_{i,s} \in H$. Show that the set of all $h_{i,s}$ generates H.*

Exercise 11 *Fill in the details from Proposition 29. In particular, show that the subshift defined by $\mathcal{G}$ is indeed $X_{\mathcal{F}}^{[R]}$.*

Exercise 12 *Let G be the subgroup of $GL_2(\mathbb{Q})$ generated by the matrices $a = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ and $b = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$. Show that every matrix in G is of the form $\begin{pmatrix} x & y \\ 0 & z \end{pmatrix}$ where x, y, z are dyadic rationals (rationals of the form $p/2^q$), and that all such matrices with $x = z = 1$ belong to G. Deduce that G contains a subgroup that is not finitely generated. Show that G is a Baumslag-Solitar group.*

Exercise 13 *Let $G = \mathbb{F}_2 = \langle a, b \rangle$ the free group generated by a and b. Consider the Cayley graph $\Gamma(G, \mathsf{S})$ for $\mathsf{S} = \{a, b, a^{-1}, b^{-1}\}$. Show that this Cayley graph can be covered by disjoints biinfinite paths (in the sense of Theorems 43 and 44).*

Exercise 14 *Let $G = PSL_2(\mathbb{Z}) = \langle a, b | a^2, b^3 \rangle$ Consider the Cayley graph $\Gamma(G, \mathsf{S})$ for $\mathsf{S} = \{a, b, a^{-1}, b^{-1}\}$ depicted below.*



Show that this Cayley graph cannot be covered by disjoints biinfinite paths (in the sense of Theorems 43 and 44). (This example shows that the choice of S *is important for this theorem to hold).*

Exercise 15 *Consider again* $G = PSL_2(\mathbb{Z}) = \langle a, b | a^2, b^3 \rangle$. *Show that* $X_{G,\{a,b\}}$ *is nonempty.*

Exercise 16 *Consider again* $G = PSL_2(\mathbb{Z}) = \langle a, b | a^2, b^3 \rangle$. *Draw the Cayley graph* $\Gamma(G, \mathsf{S})$ *for* $\mathsf{S} = \{a, ab, a^{-1}, (ab)^{-1}\}$. *Show that this Cayley graph can be covered by disjoints biinfinite paths (in the sense of Theorems 43 and 44).*

Exercise 17 *A* weak valid pair *for* (G, S) *is a pair* (n, p) *from* G *to* S *s.t.* $\mathsf{p}(gs) = s^{-1}$ *where* $s = \mathsf{n}(g)$. *Give an example for* $G = \mathbb{F}_2 = \langle a, b \rangle$ *and* $S = \{a, b\}$ *of a weak valid pair that is not a valid pair.*

Exercise 18 *Show that if* G *is finite, any weak valid pair is a valid pair.*

Exercise 19 *Give the proofs of Propositions 45 and 46.*

Exercise 20 *Give an example of a* $\mathbb{Z}$*-subshift which is effectively closed but with an undecidable language.* ***Hint****: Consider the set of forbidden words* $\{10^n 1\}_{n \in L}$ *for an appropriate set* L.

Exercise 21 *Show that the class of effectively closed subshifts is closed under finite intersections. Prove that the same result does not hold for countable intersections.*

Exercise 22 *Show that for recursively presented groups the class of effectively closed subshifts is closed under finite unions.*

References

[1] Aubrun, N., Barbieri, S., Sablik, M.: A notion of effectiveness for subshifts on finitely generated groups. Theoretical Computer Science **661**, 35–55 (2017)

[2] Aubrun, N., Kari, J.: Tiling Problems on Baumslag-Solitar groups. In: MCU'13, pp. 35–46 (2013)

[3] Aubrun, N., Sablik, M.: Multidimensional effective s-adic systems are sofic. Uniform Distribution Theory **9**(2) (2014)

[4] Ballier, A., Jeandel, E.: Tilings and model theory. First Symposium on Cellular Automata Journées Automates Cellulaires. (2008)

[5] Ballier, A., Jeandel, E.: Computing (or not) quasi-periodicity functions of tilings. In: Second Symposium on Cellular Automata "Journées Automates Cellulaires", JAC 2010, Turku, Finland, December 15-17, 2010. Proceedings, pp. 54–64 (2010)

[6] Barbieri, S., Sablik, M.: A generalization of the simulation theorem for semidirect products. Ergodic Theory and Dynamical Systems (2017)

[7] Berger, R.: The Undecidability of the Domino Problem. Ph.D. thesis, Harvard University (1964)

[8] Berger, R.: The Undecidability of the Domino Problem. No. 66 in Memoirs of the American Mathematical Society. The American Mathematical Society (1966)

[9] Blondel, V.D., Bournez, O., Koiran, P., Papadimitriou, C., Tsitsiklis, J.N.: Deciding stability and mortality of piecewise affine dynamical systems. Theoretical Computer Science **255**(1-2), 687–696 (2001)

[10] Büchi, J.R.: Turing-Machines and the Entscheidungsproblem. Math. Annalen **148**(3), 201–213 (1962). DOI 10.1007/BF01470748

[11] Ceccherini-Silberstein, T., Coornaert, M.: Cellular Automata and Groups. Springer Monographs in Mathematics. Springer-Verlag (2010)

[12] Cenzer, D., Dashti, S.A., King, J.L.F.: Computable symbolic dynamics. Mathematical Logic Quarterly **54**(5), 460–469 (2008)

[13] Cohen, D.B.: The large scale geometry of strongly aperiodic subshifts of finite type. Advances in Mathematics **308**, 599–626 (2017)

[14] Delvenne, J.C., Kůrka, P., Blondel, V.D.: Computational Universality in Symbolic Dynamical Systems, pp. 104–115. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)

[15] Diestel, R.: A short proof of Halin's grid theorem. Abh. Math. Sem. Univ. Hamburg **74**, 237–242 (2004)

[16] Durand, B., Romashchenko, A., Shen, A.: Effective Closed Subshifts in 1D Can Be Implemented in 2D, pp. 208–226. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

[17] Goodman-Strauss, C.: Matching Rules and Substitution Tilings. Annals of Mathematics **147**(1), 181–223 (1998)

[18] Goodman-Strauss, C.: A hierarchical strongly aperiodic set of tiles in the hyperbolic plane. Theoretical Computer Science **411**(7), 1085 – 1093 (2010). DOI 10.1016/j.tcs.2009.11.018

[19] Hadamard, J.: Théorème sur les séries entières. Acta Math. **22**, 55–63 (1899)

[20] Hedlund, G., Morse, M.: Symbolic dynamics. American Journal of Mathematics **60**(4), 815–866 (1938)

[21] Hochman, M.: On the dynamics and recursive properties of multidimensional symbolic systems. Inventiones Mathematicae **176**(1), 2009 (2009)

[22] Hooper, P.K.: The Undecidability of the Turing Machine Immortality Problem. Journal of Symbolic Logic **31**(2), 219–234 (1966)

[23] Jeandel, E.: Aperiodic subshifts on polycyclic groups. CoRR **abs/1510.02360** (2015). URL http://arxiv.org/abs/1510.02360

[24] Jeandel, E.: Translation-like actions and aperiodic subshifts on groups. CoRR **abs/1508.06419** (2015). URL http://arxiv.org/abs/1508.06419

[25] Jeandel, E.: Translation-like Actions and Aperiodic Subshifts on Groups (2016). ArXiv:1508.06419

[26] Jeandel, E., Vanier, P.: Characterizations of periods of multi-dimensional shifts. Ergodic Theory and Dynamical Systems **35**(2), 431–460 (2015)

[27] Kahr, A., Moore, E.F., Wang, H.: Entscheidungsproblem reduced to the $\forall\exists\forall$ case. Proceedings of the National Academy of Sciences of the United States of America **48**(3), 365–377 (1962)

[28] Kari, J.: The tiling problem revisited. In: J.O. Durand-Lose, M. Margenstern (eds.) MCU, *Lecture Notes in Computer Science*, vol. 4664, pp. 72–79. Springer (2007)

[29] Kari, J.: On the undecidability of the tiling problem. In: Current Trends in Theory and Practice of Computer Science (SOFSEM), pp. 74–82 (2008)

[30] Kari, J., Ollinger, N.: Periodicity and Immortality in Reversible Computing, pp. 419–430. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)

[31] Koiran, P., Cosnard, M., Garzon, M.: Computability with low-dimensional dynamical systems. Theoretical Computer Science **132**(1), 113 – 128 (1994)

[32] Kurka, P.: On topological dynamics of Turing machines. Theoretical Computer Science **174**, 203–216 (1997). DOI 10.1016/S0304-3975(96)00025-4

[33] Kuske, D., Lohrey, M.: Logical aspects of cayley-graphs: the group case. Annals of Pure and Applied Logic **131**(1–3), 263 – 286 (2005). DOI http://dx.doi.org/10.1016/j.apal.2004.06.002

[34] Lennox, J.C., Robinson, D.J.: The Theory of Infinite Soluble Groups. Oxford Mathematical Monographs. Oxford Science Publications (2004)

[35] Lind, D., Marcus, B.: An Introduction to Symbolic Dynamics and Coding. Cambridge University Press (1995)

[36] Lind, D.A., Marcus, B.: An Introduction to Symbolic Dynamics and Coding. Cambridge University Press (1995)

[37] Magnus, W., Karrass, A., Solitar, D.: Combinatorial Group Theory, 2nd revised edn. Dover (1976)

[38] Markley, N.G., Paul, M.E.: Matrix subshifts for $\mathbb{Z}^\nu$ Symbolic Dynamics. Proceedings of the London Mathematical Society **3**(43), 251–272 (1981)

[39] Mozes, S.: Tilings, substitutions systems and dynamical systems generated by them. J. d'Analyse Math. **53**, 139–186 (1989)

[40] Muchnik, R., Pak, I.: Percolation on Grigorchuk Groups. Communications in Algebra **29**(2), 661–671 (2001)

[41] Muller, D.E., Schupp, P.E.: The theory of ends, pushdown automata, and second-order logic. Theoretical Computer Science **37**(0), 51 – 75 (1985). DOI http://dx.doi.org/10.1016/0304-3975(85)90087-8

[42] Myers, D.: Non Recursive Tilings of the Plane II. Journal of Symbolic Logic **39**(2), 286–294 (1974)

[43] Nasu, M.: Textile Systems for Endomorphisms and Automorphisms of the Shift, *Memoirs of the American Mathematical Society*, vol. 114. American Mathematical Society (1995)

[44] Odifreddi, P.: Classical Recursion Theory, *Studies in Logic and the Foundations of Mathematics*, vol. 143. Elsevier (1999). DOI 10.1016/S0049-237X(99)80046-9

[45] Pavlov, R., Schraudner, M.: Classification of sofic projective subdynamics of multidimensional shifts of finite type. Transactions of the American Mathematical Society **367**, 3371–3421 (2015)

[46] Rips, E.: Subgroups of small cancellation groups. Bulletin of the London Mathematical Society **14**, 45–47 (1982). DOI 10.1112/blms/14.1.45

[47] Robertson, N., Seymour, P.: Graph minors. v. excluding a planar graph. Journal of Combinatorial Theory, Series B **41**(1), 92 – 114 (1986). DOI 10.1016/0095-8956(86)90030-4

[48] Robinson, R.M.: Undecidability and nonperiodicity for tilings of the plane. Inventiones Mathematicae **12**, 177–209 (1971)

[49] Robinson, R.M.: Undecidable tiling problems in the hyperbolic plane. Inventiones Mathematicae **44**, 159–264 (1978)

[50] Sablik, M., Fernique, T.: Local Rules for Computable Planar Tilings. In: Automata and Journées Automates Cellulaires 2012. . Corse, France (2012)

[51] Segal, D.: Polycyclic groups. No. 82 in Cambridge Tracts in Mathematics. Cambridge University Pres (1983)

[52] Selman, A.L.: Arithmetical reducibilities i. Mathematical Logic Quarterly **17**(1), 335–350 (1971). DOI 10.1002/malq.19710170139. URL http://dx.doi.org/10.1002/malq.19710170139

[53] Seward, B.: Burnside's Problem, Spanning Trees and Tilings. Geometry and Topology **18**, 179–210 (2014). DOI 10.2140/gt.2014.18.179

[54] Sipser, M.: Introduction to the Theory of Computation, 1st edn. International Thomson Publishing (1996)

[55] Tits, J.: Free subgroups in linear groups. Journal of Algebra **20**(2), 250–270 (1972). DOI 10.1016/0021-8693(72)90058-0

[56] Torma, I.: Quantifier extensions of multidimensional sofic shifts. Proc. Amer. Math. Soc. **143**, 4775–4790 (2015)

[57] Turing, A.: On computable numbers, with an application to the entscheidungsproblem. Proceedings of the London Mathematical Society **42**(2), 230–265 (1936)

[58] Wang, H.: Proving theorems by pattern recognition ii. Bell System Technical Journal **40**(1-3), 1–41 (1961)

[59] Whyte, K.: Amenability, Bilipschitz equivalence, and the Von Neumann Conjecture. Duke Mathematical Journal **99**(1), 93–112 (1999). DOI 10.1215/S0012-7094-99-09904-0

[60] Woess, W.: Graphs and groups with tree-like properties. Journal of Combinatorial Theory, Series B **47**(3), 361 – 371 (1989)