

# A notion of effectiveness for subshifts on finitely generated groups.

Nathalie Aubrun\*, Sebastián Barbieri\*  
and Mathieu Sablik†  
nathalie.aubrun@ens-lyon.fr  
sebastian.barbieri@ens-lyon.fr  
sablík@latp.univ-mrs.fr

## Abstract

We define a notion of effectiveness for subshifts on finitely generated groups. The set of effective subshifts forms a conjugacy class that contains the class of sofic subshifts. We prove that the inclusion is strict for several groups, including amenable groups and groups with more than two ends.

## Introduction

Symbolic dynamics were originally defined on  $\mathbb{Z}$  in the highly influential article of Morse and Hedlund [15] in order to study discretization of dynamical systems, and were later generalized to higher dimensions. Multidimensional subshifts of finite type ( $\mathbb{Z}^d$ -SFT) and sofic  $\mathbb{Z}^d$ -subshifts are the central objects in symbolic dynamics on  $\mathbb{Z}^d$ . They are sets of configurations that respect some local constraints, and can be described by a finite amount of information. When  $d \geq 2$  it turns out that they enjoy interesting computational properties, among which is the undecidability of the emptiness problem, also known as the domino problem [5, 26]. This problem can be naturally generalized to any group, nevertheless no characterization of the groups where the domino problem is undecidable is yet known even if some partial results have arisen [25, 2, 4]. This example illustrates how computational problems may depend on properties of the group considered. These latter can also modify dynamical properties, the most famous example being the property of being sofic [14, 28]. These two observations justify the study of symbolic dynamics on finitely generated groups.

More recently, the use of computability theory has become essential in the study of multidimensional subshifts of finite type. For example the possible

---

\*LIP, ENS de Lyon – CNRS – INRIA – UCBL – Université de Lyon

†Aix-Marseille Université, CNRS, Centrale Marseille, I2M UMR 7373 – 13453 Marseille, France

**Keywords:** 37B10 Symbolic dynamics, 03D10 Turing machines, 20F10 Word problems.

entropies of these systems are characterized as right recursively enumerable numbers [17]. This type of results comes from the possibility to encode Turing machines inside multidimensional SFT. The study of such results led to introduce the class of effective  $\mathbb{Z}^d$ -subshifts, defined by a recognizable set of forbidden patterns. This class was introduced by M. Hochman [16] who showed that they can be realized as subsystems of sofic  $\mathbb{Z}^{d+2}$ -subshifts. The construction was improved with two different techniques [3, 11] to get a realization in sofic  $\mathbb{Z}^{d+1}$ -subshifts. Thus with an increase of one of the dimension, effective  $\mathbb{Z}^d$ -subshifts are very close to sofic subshifts. Hochman's result suggests that if we play with the structure on which subshifts are defined, some strong links between sofic and effective subshifts may emerge.

In this direction we investigate subshifts defined on infinite finitely generated groups and define a notion of effectiveness. The difficulty for this task relies on the possibility, even for a finitely presented group, to have an undecidable word problem [21, 6] – no algorithm can decide whether a word on the generators and their inverses represents the identity element. We overcome this problem by defining a notion of effectiveness based on  $G$ -machines, that are roughly speaking Turing machines having the group  $G$  as the calculation tape.

The paper is organized as follows. The first section introduces basic notions of combinatorial group theory, including the word problem for a finitely generated group (Section 1.1), and general settings about symbolic dynamics on finitely generated groups (Section 1.2). The content of this last section is very similar to any standard introduction to classical symbolic dynamics on  $\mathbb{Z}$  [18] or  $\mathbb{Z}^d$ . Nonetheless as there is almost no literature treating symbolic dynamics in this setting it is necessary to state the definitions and fix the notations. In Section 2 we present two definitions of effectiveness for  $G$ -subshifts:  $\mathbb{Z}$ -effectiveness and  $G$ -effectiveness, that coincide if and only if the finitely generated group  $G$  considered has decidable word problem (Theorem 2.10). Section 3 exhibits classes of groups with  $G$ -effective subshifts which are not sofic. This is the case for the three following classes of finitely generated groups:

1. recursively presented groups with undecidable word problem – Theorem 3.1,
2. infinite amenable groups – Theorem 3.4,
3. groups which have two or more ends – Theorem 3.6.

Particularly, we prove that the  $G$ -subshift  $X_{\leq 1}$  on  $\{0, 1\}$  consisting of configurations that contain at most one symbol 1 cannot be sofic in the first case. This example complements results of [10] where the property of  $X_{\leq 1}$  being sofic is related to geometric properties of the group, and answers an open question [9, 29].

## 1 Generalities

### 1.1 Finitely generated groups and computational aspects

Let  $G$  be a group and  $1_G$  be its identity element. The group  $G$  is *finitely generated* if there is a finite subset  $S \subset G$  which generates  $G$  or equivalently if it

has a presentation such that  $G \simeq \langle S|R \rangle$  with  $|S| < \infty$  (see [19] for an introduction to group presentations). In a presentation  $\langle S|R \rangle$ ,  $S$  is the set of *generators* of the group, and  $R$  is the set of *relators* and is made of words on the alphabet  $S \cup S^{-1}$  (where  $S^{-1}$  the set of inverses of generators) that represent the identity of the group. A group has infinitely many presentations and determining whether two presentations define two isomorphic groups is an undecidable problem [24].

Let  $G$  be a group generated by a finite set  $S \subset G$ . Two words  $u, v$  in  $(S \cup S^{-1})^*$  are *equivalent in  $G$* , and this equivalence is denoted  $u =_G v$ , if  $u$  and  $v$  are equal as elements of  $G$ . We use words to represent elements of the group if the context is clear enough.

**Example** For  $BS(1, 2)$  the Baumslag-Solitar group with presentation  $\langle a, b | ab = ba^2 \rangle$ , the words  $abab^{-1}$  and  $bab^{-1}a$  are equivalent since we have the following:

$$bab^{-1}a = bab^{-1}(ab)b^{-1} = bab^{-1}(ba^2)b^{-1} = (ba^2)ab^{-1} = abab^{-1}.$$

Let  $G$  be a group and  $1_G \notin S \subset G$ . The *Cayley graph* of  $G$  given by  $S$ , denoted by  $\Gamma(G, S) = (V_\Gamma, E_\Gamma)$ , is an undirected vertex transitive graph such that  $V_\Gamma = G$  and  $E_\Gamma = \{\{g, gs\} \mid g \in G, s \in S\}$ . This graph is usually defined as directed but for our purposes it suffices to consider it as a standard graph. If  $S$  is a finite set of generators of  $G$  then  $\Gamma(G, S)$  is connected and locally finite. For  $g \in G$  we denote  $|g|$  the length of the shortest path from  $1_G$  to  $g$  in  $\Gamma(G, S)$ . We also define the *ball of size  $n \geq 0$*  as  $B_n = \{g \in G \mid |g| \leq n\}$ . Also for  $F \subset G$  a finite set we define its *interior*  $\mathring{F} := \{g \in F \mid \forall s \in S, gs \in F\}$  and its *boundary*  $\partial F := F \setminus \mathring{F}$ .

Naturally, the definitions above depend on the choice of generating set  $S$ , nevertheless all the metrics generated by the distance in  $\Gamma(G, S')$ , where  $S'$  is a finite set of generators, are equivalent.

**Example** The Cayley graph of  $\mathbb{Z}^2$  with presentation  $\langle a, b | ab = ba \rangle$  is the bi-infinite grid. The Cayley graph of the free group with two generators  $\langle a, b | \emptyset \rangle$  is the 4-regular infinite tree.

We use some basic concepts of computability theory, a good introduction can be found in [1]. We recall here two fundamental notions. Let  $\mathcal{A}^*$  be the set of all words over a finite alphabet  $\mathcal{A}$ . A subset  $L \subset \mathcal{A}^*$  is *decidable* if there exists a Turing machine that accepts if a sequence  $w \in \mathcal{A}^*$  is in  $L$  and rejects otherwise. A subset of  $L \subset \mathcal{A}^*$  is *recognizable* if there is a Turing machine that lists its elements (in no particular order). It is equivalent to say that there exists a Turing machine that accepts when a sequence  $w \in \mathcal{A}^*$  is in  $L$ , and may give no answer otherwise. A language  $L$  is decidable if and only if both  $L$  and its complement are recognizable (we say that  $L$  is *co-recognizable*).

Given a group  $G$  with a finite set of generators  $S \subset G$ , the *word problem* of  $G$  asks whether two words on  $S \cup S^{-1}$  are equivalent in  $G$ . In other terms, is there a Turing machine that decides whether two words  $w_1, w_2 \in (S \cup S^{-1})^*$  satisfy  $w_1 =_G w_2$ . We adopt here the notation:

$$WP(G) = \left\{ w \in (S \cup S^{-1})^* \mid w =_G 1_G \right\}.$$

The word problem can thus be reformulated as: is  $WP(G)$  decidable? The decidability of the word problem is independent of the set of generators chosen for  $G$ , thus the notation  $WP(G)$  is appropriate. A fundamental result of Novikov [21] and Boone [6] exhibits finitely presented groups (the set of relators is also finite) with undecidable word problem.

## 1.2 Symbolic dynamics and subshifts

Let  $\mathcal{A}$  be a finite alphabet and  $G$  a group. We say that the set  $\mathcal{A}^G = \{x : G \rightarrow \mathcal{A}\}$  equipped with the left group action  $\sigma : G \times \mathcal{A}^G \rightarrow \mathcal{A}^G$  such that  $(\sigma_g(x))_h = x_{g^{-1}h}$  is the  $G$ -fullshift. The elements  $a \in \mathcal{A}$  and  $x \in \mathcal{A}^G$  are called *symbols* and *configurations* respectively.

By taking the discrete topology on  $\mathcal{A}$  we obtain by Tychonoff's theorem that the product topology in  $\mathcal{A}^G$  is compact. This topology is generated by a clopen basis given by the *cylinders*  $[a]_g = \{x \in \mathcal{A}^G \mid x_g = a \in \mathcal{A}\}$ . If  $G$  is countable, then  $\mathcal{A}^G$  is metrizable and the compactness of the product topology can be proven directly without using Tychonoff's theorem. In the case of a finitely generated group  $G$ , an ultrametric which generates the product topology is given by  $d(x, y) = 2^{-\inf\{|g| \mid g \in G : x_g \neq y_g\}}$ .

**Definition** A subset  $X$  of  $\mathcal{A}^G$  is a  $G$ -subshift if it is  $\sigma$ -invariant –  $\sigma(X) \subset X$  – and closed for the cylinder topology.

**Example** Let  $G$  be a group and consider the alphabet  $\{0, 1\}$ . Define  $X_{\leq 1}$  as the set of configurations that contain at most one symbol 1.

$$X_{\leq 1} = \{x \in \{0, 1\}^G \mid |\{g \in G : x_g = 1\}| \leq 1\}.$$

One can easily check that  $X_{\leq 1}$  is both closed and  $\sigma$ -invariant. Thus  $X_{\leq 1}$  is a  $G$ -subshift, called the *one-or-less subshift*.

A *support* is a finite subset  $F \subset G$ . Given a support  $F$ , a *pattern with support*  $F$  is an element  $P$  of  $\mathcal{A}^F$ , i.e. a finite configuration and we write  $\text{supp}(P) = F$ . One says that a pattern  $P \in \mathcal{A}^F$  *appears* in a configuration  $x \in \mathcal{A}^G$  if there exists  $g \in G$  such that for any  $h \in F$ ,  $x_{gh} = P_h$ , in this case we write  $P \sqsubset x$ . We denote the set of finite patterns over  $G$  as  $\mathcal{A}_G^* := \bigcup_{F \subset G, |F| < \infty} \mathcal{A}^F$ . For  $P \in \mathcal{A}_G^*$  and  $g \in G$  the *cylinder generated by  $P$  on  $g$*  is  $[P]_g := \bigcap_{h \in \text{supp}(P)} [P_h]_{gh}$ . For a  $G$ -subshift  $X \subset \mathcal{A}^G$  the *set of patterns of support  $F$*  is  $\mathcal{L}_F(X) := \{P \in \mathcal{A}^F \mid \exists x \in X, P \sqsubset x\}$  and the *language* of  $X$  is  $\mathcal{L}(X) := \bigcup_{F \subset G, |F| < \infty} \mathcal{L}_F(X)$ .

By using the cylinder nature of patterns it is easy to show the following combinatorial characterization of  $G$ -subshifts:

**Proposition 1.1.** *A subset  $X$  of  $\mathcal{A}^G$  is a  $G$ -subshift if and only if there exists a set of forbidden patterns  $\mathcal{F} \subset \mathcal{A}_G^*$  that defines it.*

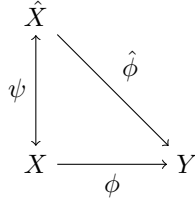
$$X = X_{\mathcal{F}} := \{x \in \mathcal{A}^G \mid \forall P \in \mathcal{F}, P \not\sqsubset x\}.$$

Let  $X, Y$  be two  $G$ -subshifts over alphabets  $\mathcal{A}_X, \mathcal{A}_Y$  and  $F$  a finite subset of  $G$ . We say that  $\phi : X \rightarrow Y$  is a *sliding block code* if there exists a local function  $\Phi : \mathcal{A}_X^F \rightarrow \mathcal{A}_Y$  such that  $F \subset G$  is finite and  $\phi(x)_g := \Phi(\sigma_{g^{-1}}(x)|_F)$ , that is denoted  $\phi = \Phi_\infty$ . A famous theorem by Curtis, Lyndon and Hedlund – see for example [7] – identifies the class of sliding block codes – called cellular automata in the reference – with the class of continuous shift commuting functions. We say that a sliding block code  $\phi$  is a *factor code* if it is surjective, and we say it is a *conjugacy* if it is bijective.

Whenever there is a factor code  $\phi : X \twoheadrightarrow Y$  we will write  $X \twoheadrightarrow Y$  and say that  $Y$  is a *factor* of  $X$  and that  $X$  is an *extension* of  $Y$ . Furthermore, if  $\phi$  is a conjugacy we will write  $X \simeq Y$  and say they are *conjugated*. The conjugacy is an equivalence relation which preserves most of the topological dynamics of a system.

**Example** Let  $G$  be a group and  $X \subset \mathcal{A}^G$  a  $G$ -subshift. Given a finite subset  $1_G \in F \subset G$  consider the sliding block code  $\beta_F : X \rightarrow (\mathcal{L}_F(X))^G$  where  $(\beta_F(x))_g = \sigma_{g^{-1}}(x)|_F \in \mathcal{L}_F(X)$ . Denote  $X^{[F]} := \beta_F(X)$  the *F-higher block presentation* of  $X$ . We claim  $\beta_F$  is a conjugacy. Indeed, by defining  $\phi : X^{[F]} \rightarrow X$  with  $\phi = \Phi_\infty$  where  $\Phi(F) = F_{1_G}$  we observe that  $\phi \circ \beta_F = id_X$  and so  $\beta_F$  is a conjugacy and  $X \simeq X^{[F]}$ .

If  $\phi : X \rightarrow Y$  is a sliding block code defined by a local function  $\Phi : \mathcal{A}_X \rightarrow \mathcal{A}_Y$  then we will say that  $\phi$  is a *1-block code*. For every sliding block code  $\phi : X \rightarrow Y$  it is possible to find a conjugacy  $\psi : X \rightarrow \hat{X}$  and a 1-block code  $\hat{\phi} : \hat{X} \rightarrow Y$  such that  $\phi = \hat{\phi} \circ \psi$ . This means that for every extension of a given  $G$ -subshift  $Y$  we can ask for a conjugate version  $\hat{X}$  of  $X$  which extends  $Y$  by a 1-block code. To see this, note that if  $\Phi$  is defined over  $\mathcal{A}_X^F$ , then  $\hat{X} := X^{[F]} \simeq X$  and  $\hat{\Phi} = \Phi$  is now a 1-block code.



We say that a  $G$ -subshift  $X \subset \mathcal{A}^G$  is a *G-subshift of finite type* –  $G$ -SFT for short – if it can be defined by a finite set of forbidden patterns, that is,  $|\mathcal{F}| < \infty$  and  $X = X_{\mathcal{F}}$ . We say that a  $G$ -subshift  $Y$  is *sofic* if there exists a  $G$ -SFT  $X$  and a factor code  $\phi$  such that  $\phi(X) = Y$ . The class of sofic  $G$ -subshifts is the smallest class closed under factor codes that contains every  $G$ -SFT. Both classes are conjugacy invariants, that is, the property of belonging to them is preserved under conjugacy.

**Example** Let  $G$  be a group  $S \subset G$  a finite set of generators. The generalized *S-Fibonacci shift* is the set  $X_{fib} = X_{\mathcal{F}} \subset \{0, 1\}^G$  such that  $P \in \mathcal{F}$  if and only if  $supp(P) = \{1_g, s\}$  with  $s \in S$  and  $P_{1_G} = P_s = 1$ . The  $S$ -Fibonacci shift is a

$G$ -SFT for every finitely generated group  $G$ . The name comes from the original definition where  $G = \mathbb{Z}$  and  $S = \{+1\}$  as

$$|\mathcal{L}_{[0,n]}(X_{fib})| = |\mathcal{L}_{[0,n-1]}(X_{fib})| + |\mathcal{L}_{[0,n-2]}(X_{fib})|.$$

**Example** The subshift  $X_{\leq 1}$  is not a  $G$ -SFT for every infinite group  $G$ . If it were consider the finite set  $K := \bigcup_{P \in \mathcal{F}} \text{supp}(P)$  and as both the pattern consisting only of 0, and the one containing a single 1 surrounded by 0 are in  $\mathcal{L}(X_{\leq 1})$  then by choosing  $g \in G$  such that  $g(K^2) \cap (K^2) = \emptyset$  (where  $K^2 := \{k_1 k_2 \mid k_1, k_2 \in K\}$ ) then  $x \in \{0, 1\}^G$  such that  $x_{1_G} = x_g = 1$  and  $x_h = 0$  otherwise, contains no forbidden patterns.

**Example** Consider  $F_k$  the free group on  $k \geq 1$  generators. The  $F_k$ -subshift  $X_{\leq 1}$  is sofic. Indeed, suppose  $S = \{s_1, \dots, s_k\}$  the generators of  $F_k$  and consider  $\mathcal{B} := B_1$ . We construct  $X = X_{\mathcal{F}} \subseteq \mathcal{B}^{F_k}$  which is an  $F_k$ -SFT extension of  $X_{\leq 1}$ . The set of forbidden patterns  $\mathcal{F}$  is given by all  $P$  such that  $\text{supp}(P) = \{1_G, s\}$  with  $s \in S$  and satisfying that either  $P_{1_G} = 1_G$  and  $P_s \neq s$  or  $P_{1_G} = s'$ ,  $P_s \neq s$  and  $s \neq s'^{-1}$ . By projecting with  $\phi = \Phi_{\infty}$  where  $\Phi(1_G) = 1$  and  $\Phi(B_1 \setminus \{1_G\}) = 0$  then  $\phi(X) = X_{\leq 1}$ .

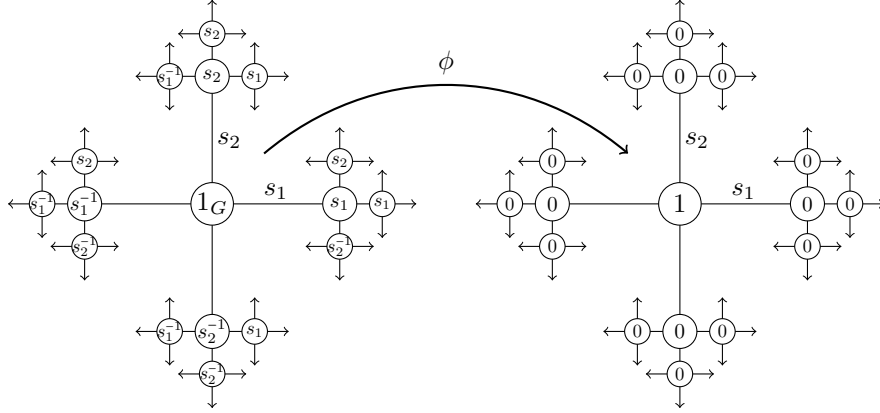


Figure 1: The extension showing that  $X_{\leq 1}$  is  $F_2$ -sofic.

Let  $G$  be a group generated by a finite set  $S \subset G$ . We say a  $G$ -SFT is  $S$ -nearest neighbor if there is a finite set of forbidden patterns  $\mathcal{F}$  such that  $X = X_{\mathcal{F}}$  and every  $P \in \mathcal{F}$  has support  $\text{supp}(P) = \{1_G, s\}$  where  $s \in S$ . Not every  $G$ -SFT is  $S$ -nearest neighbor, but every  $G$ -SFT admits a conjugated version which satisfies the property.

Indeed, if  $X \subset \mathcal{A}^G$  is a  $G$ -SFT then  $X = X_{\mathcal{F}}$  for a finite set of forbidden patterns  $\mathcal{F}$ . Consider  $F = \bigcup_{P \in \mathcal{F}} \text{supp}(P)$  and  $X^{[F]} \simeq X$  the  $F$ -higher block representation of  $X$ . We claim that  $X^{[F]} = X_G$  where  $P \in \mathcal{G}$  if and only if

$\text{supp}(P) \in \{1_G, s\}$  with  $s \in S$  and  $P_{1_G}|_{F \cap sF} \neq P_s|_{s^{-1}F \cap F}$ . Just note that  $y \in X^{[F]}$  if and only if  $\exists x \in X$  such that  $y_g = \sigma_{g^{-1}}(x)|_F$  and thus

$$y_g|_{F \cap sF} = \sigma_{g^{-1}}(x)|_{F \cap sF} = \sigma_{s^{-1}g^{-1}}(x)|_{s^{-1}F \cap F} = y_{gs}|_{s^{-1}F \cap F}.$$

As  $S$  is finite, then  $\mathcal{G}$  is a finite set of forbidden patterns and thus  $X^{[F]}$  is a  $S$ -nearest neighbor  $G$ -SFT.

By mixing the proofs of the previous results we obtain the following property that will be used many times in this work.

**Proposition 1.2.** *Let  $G$  be a group generated by a finite set  $S$ , then for every sofic  $G$ -subshift  $Y$  there is a  $S$ -nearest neighbor  $G$ -SFT extension  $X$  given by a 1-block factor code  $\phi: X \twoheadrightarrow Y$ .*

## 2 Effectiveness on finitely generated groups

We say a  $\mathbb{Z}$ -subshift  $X \subset \mathcal{A}^{\mathbb{Z}}$  is *effective* if there is a recognizable set of forbidden patterns  $\mathcal{F} \subset \mathcal{A}^*$  such that  $X = X_{\mathcal{F}}$ . Equivalently, a  $\mathbb{Z}$ -subshift is effective if it can be written as the complement of a computable union of cylinders. We intend to generalize this definition to the class of finitely generated groups. On  $\mathbb{Z}^d$ , a finite pattern is no longer a word, but it can be easily coded as a word – via any recursive bijection between  $\mathbb{Z}^d$  and  $\mathbb{Z}$  – then effective  $\mathbb{Z}^d$ -subshifts correspond to subshifts which can be defined by a set of forbidden patterns that admits a recognizable coding. In general groups, this recursive bijection might not exist.

In this section we first take the previous ideas of codings to the context of finitely generated groups and explore their limitations with regards to the word problem of the group. Next we define a general notion of effectiveness which has better properties than the interpretation via codings. Finally we show where these two effectiveness notions match, exhibit some stability properties for the classes they define and compare them with sofic  $G$ -subshifts and  $G$ -SFTs.

### 2.1 $\mathbb{Z}$ -Effectiveness

Let  $G$  be a group generated by a finite set  $S \subset G$  and  $\mathcal{A}$  a finite alphabet. A *pattern coding*  $c$  is a finite set of tuples  $c = (w_i, a_i)_{1 \leq i \leq n}$  with  $n \in \mathbb{N}$  where  $w_i \in (S \cup S^{-1})^*$  and  $a_i \in \mathcal{A}$ . We say that a pattern coding is *consistent* if for every pair of tuples such that  $w_i =_G w_j$  then  $a_i = a_j$ . For a consistent pattern coding  $c$  we define the pattern  $\Pi(c) \in \mathcal{A}_G^*$  such that  $\text{supp}(\Pi(c)) = \bigcup_{i \in I} w_i$  and  $\Pi(c)_{w_i} = a_i$ .

**Example** Let  $G$  be the Baumslag-Solitar group  $BS(1, 2)$  given by the finite presentation  $\langle a, b \mid ab = ba^2 \rangle$ , and consider the finite alphabet  $\mathcal{A} = \{0, 1\}$ . Then the pattern coding

$$\begin{array}{ccc} (\epsilon, 0) & (b, 1) & (a, 1) \\ (ab, 0) & (ba^2, 0) & (ba, 1) \end{array}$$

is consistent, since all the words above on  $\{a, b, a^{-1}, b^{-1}\}$  represent different elements in  $G$  except for  $ab$  and  $ba^2$  that are assigned the same symbol 0. The pattern  $\Pi$  it defines is:

$$\begin{array}{ccccc} \Pi_{1_G}=0 & & & & \Pi_a=1 \\ & \frown & & \smile & \\ & | & & | & \\ \Pi_b=1 & \Pi_{ba}=1 & & \Pi_{ba^2}=\Pi_{ab}=0 & \end{array}$$

But the pattern coding

$$\begin{array}{ccc} (\epsilon, 0) & (a^2, 1) & (bab^{-1}a, 1) \\ (a, 1) & (ba, 1) & (abab^{-1}, 0) \end{array}$$

is inconsistent since words  $abab^{-1}$  and  $bab^{-1}a$  represent the same element in  $G$  but are assigned different symbols.

The specific choice of the set of generators  $S$  is irrelevant as one can easily traduce one in terms of the other. In order to recognize whether a pattern codified as above belongs to a given set of forbidden patterns it is first necessary to recognize if the pattern coding is consistent. A finitely generated group is said to be *recursively presented* if there is a presentation  $G \simeq \langle S, R \rangle$  such that  $|S| < \infty$  and  $R$  is a recognizable set.

**Proposition 2.1.** *Let  $G$  be a finitely generated group and  $\mathcal{A}$  be an alphabet with at least two symbols. The following are equivalent:*

1.  $G$  is recursively presented.
2. The word problem of  $G$  is recognizable.
3. The set of inconsistent pattern codings is recognizable.

*Proof.* The equivalence between the two first statements is trivial. Let  $G$  have recognizable word problem. As  $w_i =_G w_j \Leftrightarrow w_i(w_j)^{-1} =_G 1_G$  then checking whether  $w_i =_G w_j$  is recognizable and so the inconsistency of the pattern codings is recognizable. Conversely for input  $w$  in order to recognize if  $w =_G 1_G$  it suffices to give as input to the machine deciding the inconsistency of the pattern codings the coding  $\{(\epsilon, a), (w, b)\}$  with  $a \neq b \in \mathcal{A}$ .  $\square$

**Definition** We say a  $G$ -subshift  $X \subset \mathcal{A}^G$  is  $\mathbb{Z}$ -effective if there exist a finite set of generators  $S$  of  $G$ , a set of forbidden patterns  $\mathcal{F} \subset \mathcal{A}_G^*$  such that  $X = X_{\mathcal{F}}$  and a Turing machine  $T$  that accepts a pattern coding  $c$  if and only if it is either inconsistent or  $\Pi(c) \in \mathcal{F}$ .

**Proposition 2.2.** *In the definition above it is possible to choose  $\mathcal{F}$  to be a maximal – for inclusion – set of forbidden patterns.*



*Proof.* Suppose we are given a Turing machine  $T$ , that defines a  $\mathbb{Z}$ -effective subshift as above, and a pattern coding  $c = (w_i, a_i)_{1 \leq i \leq k}$ . Note that it is possible to see if a translation  $\sigma_g(P)$  of a pattern  $P := \Pi(c)$  is in  $\mathcal{F}$ : enumerate each word  $u \in (S \cup S^{-1})^n$  and run  $T$  up to  $n$  steps on the pattern coding  $(uw_i, a_i)_{1 \leq i \leq k}$  of  $uP$ . If the procedure accepts on an input  $(uw_i, a_i)_{1 \leq i \leq k}$ , accept the pattern  $P$ . If not, iterate with  $n \leftarrow n + 1$ . It is also easy to see if  $P$  contains a forbidden pattern: run the previous algorithm with every subset of the pattern coding.

Also, by enumerating all  $m \in \mathbb{N}$  and possible extensions of  $c$  having for support all the words in  $(S \cup S^{-1})^m$  and accepting if every extension of a given length is accepted (one of the translations of them contains a forbidden pattern) it is possible to detect if  $\Pi(c)$  cannot be extended to an infinite configuration.  $\square$

The notion of  $\mathbb{Z}$ -effectiveness does not work very well on groups which are not recursively presented. In fact, the recognizability of inconsistent patterns is equivalent to the group being recursively presented by Proposition 2.1. This means that for groups which are not recursively presented even the  $G$ -fullshift on two symbols is not  $\mathbb{Z}$ -effective. Furthermore, the class of  $\mathbb{Z}$ -effective subshifts consists uniquely of the empty subshift and subshifts on one symbol.

Even if this definition is restrained to the class of recursively presented groups, there are simple  $G$ -subshifts which are not  $\mathbb{Z}$ -effective. Recall that the one-or-less subshift  $X_{\leq 1}$  defined in Section 1.2 is the set of configurations on alphabet  $\{0, 1\}$  containing at most one symbol 1.

**Proposition 2.3.** *Let  $G$  be a finitely generated group with undecidable word problem. Then  $X_{\leq 1}$  is not  $\mathbb{Z}$ -effective.*

*Proof.* As  $\{0, 1\}$  has two symbols, the case where  $G$  is not recursively presented is deduced from Proposition 1.2, thus we suppose  $G$  is recursively presented.

Let  $G$  be generated by the finite set  $S$ . We proceed by contradiction by showing that if  $X_{\leq 1}$  is  $\mathbb{Z}$ -effective then the word problem is decidable in  $G$ . As  $G$  is recursively presented, the word problem is already recognizable. It suffices to show it is co-recognizable.

Let  $\mathcal{F}$  be a maximal set of forbidden patterns with  $X_{\leq 1} = X_{\mathcal{F}}$ . Given  $w \in (S \cup S^{-1})^*$  consider the pattern coding  $c = \{(\epsilon, 1), (w, 1)\}$ . Then  $P := \Pi(c)$  is the pattern such that  $P|_G = P_w = 1$ . If  $w \neq_G 1_G$  then the pattern coding is consistent and  $P \in \mathcal{F}$  and so the Turing machine  $T$  – given by  $\mathbb{Z}$ -effectiveness of  $X_{\leq 1}$  – accepts. Conversely if  $w =_G 1_G$  then  $T$  does not accept as the pattern coding is consistent and  $P \in \mathcal{L}(X)$ . This means the word problem is also co-recognizable and thus decidable.  $\square$

## 2.2 $G$ -Effectiveness

The class of  $\mathbb{Z}$ -effective  $G$ -subshifts does not work very well for groups with undecidable word problem, as shown by the example of  $X_{\leq 1}$ .

We propose an alternative definition of effectiveness which makes sense even for groups which are not recursively presented, and we show that it coincides

with  $\mathbb{Z}$ -effectiveness for groups with decidable word problem. The idea is to adapt the classical model of Turing machines, which receives a finite word as input, so that it receives a pattern  $P \in \mathcal{A}_G^*$  as input. To do so, we replace the classical tape by a finitely generated group  $G$ . Turing machines using Cayley graph as tape have already been mentioned in [12] and studied in depth in [8], but these latter machines take as input a word  $w \in \mathcal{A}^*$  and not a pattern  $P \in \mathcal{A}_G^*$ .

**Definition** Let  $G$  be a finitely generated group. A  $G$ -machine is a 7-tuple  $(Q, \Sigma, \sqcup, q_0, F, S, \delta)$  where  $Q$  is a finite set of states,  $\Sigma$  is a finite alphabet,  $\sqcup \in \Sigma$  is the blank symbol,  $q_0 \in Q$  is the initial state,  $F \subset Q$  is the set of final states – accepting or rejecting states –  $S$  is a finite set such that  $G$  is generated by  $S$  and  $\delta: Q \times \Sigma \rightarrow Q \times \Sigma \times (S \cup S^{-1} \cup \{1_G\})$  is the transition function.

A  $G$ -machine  $M$  is a Turing machine whose bi-infinite tape has been identified as the set of integers  $\mathbb{Z}$  and replaced by a finitely generated group  $G$ . A *configuration* of  $M$  is a tuple  $(q, c)$  in  $Q \times \Sigma^G$  such that there is at most a finite set of  $g \in G$  such that  $c_g \neq \sqcup$ . The calculation proceeds as in a usual Turing machine except that the computation head writes over the group  $G$ , and moves can be made in any direction of  $\mathcal{S} := S \cup S^{-1}$ . For the definition of calculation we adopt the convention that the computation head is always located at the identity of the group  $1_G$  and that the whole tape is shifted, though this can be adapted to suit the occasion.

The  $G$ -machine *computes* configuration  $(q', c')$  from configuration  $(q, c)$  if state  $q'$  and tape  $c'$  are given by  $\delta(q, c_{1_G}) = (q', b, s)$  where tape  $c'$  is obtained by putting a symbol  $b$  at the identity in  $c$  and applying the shift  $\sigma_{s^{-1}}$ . In this case we denote

$$(q, c) \xrightarrow{\delta(q, c_{1_G}) = (q', b, s)} (q', c')$$

or  $(q, c) \longrightarrow (q', c')$  for short (see Figure 2). For a pattern  $P \in \mathcal{A}_G^*$  denote  $c_P$  configuration uniformly filled with blank symbol  $\sqcup$  except on  $\text{supp}(P)$  where  $P$  appears. We say that a  $G$ -machine *accepts* (resp. *rejects*) a pattern  $P \in \mathcal{A}_G^*$  if starting from the initial configuration  $(q_0, c_P)$  the machine computes successively  $(q_0, c_P) \longrightarrow (q_1, c_1) \longrightarrow \dots \longrightarrow (q_n, c_n)$  and reaches in a finite number of steps a configuration with an accepting state  $q_n \in F$  (resp. rejecting state  $q_n \in F$ ).

$G$ -decidable and  $G$ -recognizable languages  $L \subset \Sigma_G^*$  are defined analogously to the standard case.

**Proposition 2.4.** *Let  $G$  be a finitely generated group and  $T_S$  a  $G$ -machine recognizing a language  $L \subset \mathcal{A}_G^*$  while using a finite set of generators  $S$ . If  $S'$  is another finite set of generators, there is another  $G$ -machine  $T_{S'}$  using  $S'$  such that the language recognized by  $T_{S'}$  is  $L$ .*

*Proof.* Each  $g \in G$  can be written as  $g = h_1 \dots h_{n(g)}$  where every  $h_i \in S' \cup S'^{-1}$ . Consider  $T_{S'}$  a copy of  $T_S$  where for each state  $q \in Q$  we add a copy  $q_{s_i}$  for  $s \in S \cup S^{-1}$  and  $i \in \{1, \dots, n(s)\}$  and every instruction  $\delta(q, a) = (p, b, s)$  in  $T_S$  is replaced with the instructions:  $\delta(q, a) = (p_{s_1}, b, h_1)$  and  $\delta(p_{s_i}, *) = (p_{s_{i+1}}, *, h_{i+1})$

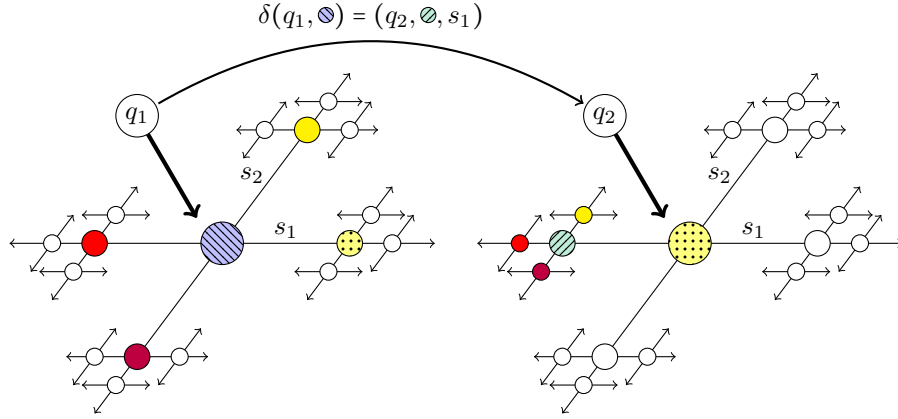


Figure 2: A transition in an  $F_2$ -machine.

for  $1 \leq i < n(s)$  where  $*$  is an arbitrary symbol in  $\mathcal{A}$  and  $\delta(p_{s_n(s)}, *) = (p, *, 1_G)$ . The modified machine  $T_{S'}$  moves with the set of generators  $S'$  and has the same output as  $T_S$ .  $\square$

The previous proposition expresses the fact that the choice of generating set is irrelevant for the computation. The class of  $G$ -machines shares also the robustness of Turing machines with respect to changes in its definition. For example, we can allow multiple tapes with multiple independent writing heads.

**Definition** Let  $G$  be a finitely generated group. A *multiple head  $G$ -machine* is the same as a  $G$ -machine, except that the machine uses  $G^n$  as a tape and the transition function is  $\delta : Q^n \times \Sigma^n \rightarrow Q^n \times \Sigma^n \times (S \cup S^{-1} \cup \{1_G\})^n$ , where  $n$  is the number of heads of the machine.

A multiple head  $G$ -machine *accepts* (resp. *rejects*) a pattern  $P \in \mathcal{A}_G^*$  if starting from the initial configuration  $(q_0^n, (c_P, \sqcup^G, \dots, \sqcup^G))$  the machine reaches in a finite number of steps a configuration with an accepting state  $q_n \in F$  (resp. rejecting state  $q_n \in \bar{F}$ ) on one of its heads.

We suppose that each computation head works on its own tape, but it can read the content of other tapes. By codifying independent movements of a tape accordingly, we are able to read not only what each head is looking at a certain step but what is written in the other tape in the group position which would correspond to another head if we considered that the heads moved in each layer and the groups don't move.

**Proposition 2.5.** *Let  $L$  be a language that can be decided by a multiple head  $G$ -machine. Then  $L$  can be decided by a  $G$ -machine*

*Proof.* The proof of this result can be found in Appendix A.  $\square$

Now we use  $G$ -machines to give a more natural definition of effectiveness in a finitely generated group.

**Definition** A  $G$ -subshift  $X \subset \mathcal{A}^G$  is  $G$ -effective if there exists a set of forbidden patterns  $\mathcal{F} \subset \mathcal{A}_G^*$  such that  $X = X_{\mathcal{F}}$  and  $\mathcal{F}$  is  $G$ -recognizable.

Next we show that this definition extends the notion of  $\mathbb{Z}$ -effectiveness.

**Theorem 2.6.** *Let  $G$  be an infinite and finitely generated group and  $X \subset \mathcal{A}^G$  a  $\mathbb{Z}$ -effective  $G$ -subshift. Then  $X$  is  $G$ -effective.*

*Proof.* Let  $T$  be a Turing machine which recognizes inconsistent pattern codings and the ones that codify patterns in  $\mathcal{F} \subset \mathcal{A}_G^*$  such that  $X = X_{\mathcal{F}}$ . We construct a  $G$ -machine  $\mathcal{M}$  which reads input  $P \in \mathcal{A}_G^*$  and accepts if and only if  $P \in \mathcal{F}$ . By Proposition 2.2 we can assume without loss of generality that  $\mathcal{F}$  is a maximal set of forbidden patterns. Moreover we can also assume that  $T$  is a one-sided Turing machine with a reading tape and a working tape.

The construction is a multiple head  $G$ -machine which consists of the following five layers (see Figure 3):

1. A reading layer where the input  $P \in \mathcal{A}_G^*$  is stored.
2. A machine  $\mathcal{M}_{\text{PATH}}$  which constructs an arbitrarily long one-sided non-intersecting path starting from  $1_G$ .
3. A machine  $\mathcal{M}_{\text{VISIT}}$  which is able to visit iteratively all the elements of  $B_n$  for  $n \in \mathbb{N}$  starting with  $n$  initially assigned to 1 ( $n \leftarrow 1$ ).
4. A layer which serves as a nexus between the first layer and the fifth.
5. A layer which simulates  $T$  in the one-sided path created by  $\mathcal{M}_{\text{PATH}}$ .

We begin by describing  $\mathcal{M}_{\text{PATH}}$ . Let  $S \subset G$  be a finite set of generators, let  $\mathcal{S} := S \cup S^{-1} = \{g_1, g_2, \dots, g_k, g_{k+1}, \dots, g_{2k}\}$  where  $g_{k+i} = g_i^{-1}$  and consider the  $G$ -machine  $\mathcal{M}_{\text{PATH}} := (Q, \Sigma, \sqcup, q_0, F, S, \delta)$  where  $Q := \{I, B\} \cup (\mathcal{S} \times \{\leftarrow, \rightarrow\})$ ,  $\Sigma = (\{\sqcup, \triangleright\} \cup \mathcal{S}) \times \{\sqcup, \times\} \times (\{\sqcup\} \cup \mathcal{S})$ ,  $q_0 = I$ ,  $F = \emptyset$  (we force the machine to loop), and  $\delta$  is given by the following rules where  $*$  stands for an arbitrary fixed symbol.

$$\begin{aligned} \delta((\sqcup, \sqcup, \sqcup), I) &= ((\triangleright, \times, g_1), g_1^{\leftarrow}, g_1). \\ \delta((\sqcup, \sqcup, \sqcup), g_i^{\leftarrow}) &= ((g_i, \times, \sqcup), g_1^{\rightarrow}, 1_G). \\ \delta((*_1, \times, *_2), g_i^{\rightarrow}) &= ((*_1, \times, g_i), g_i^{\leftarrow}, g_i). \\ \delta((*_1, \times, *_2), g_i^{\leftarrow}) &= ((*_1, \times, *_2), B, g_i^{-1}). \\ \delta((g_j, \times, g_i), B) &= \begin{cases} ((g_j, \times, g_i), g_{i+1}^{\rightarrow}, 1_G), & \text{if } i < 2k \\ ((\sqcup, \sqcup, \sqcup), B, g_j^{-1}), & \text{if } i = 2k. \end{cases} \end{aligned}$$

$$\delta((\triangleright, \times, g_i), B) = ((\triangleright, \times, g_i), g_{i+1}^{\rightarrow}, 1_G), \text{ if } i < 2k$$

The rules from  $\delta$  codify a backtracking in  $G$  which marks a one-sided non-intersecting infinite path in  $G$ . The states  $I$  and  $B$  stand for initialization and backtracking respectively. The elements from  $\Sigma$  are triples  $(a_1, a_2, a_3)$  which indicate the following information: My left and right neighbors are  $a_1$  and  $a_2$  respectively and I belong to the path if  $a_2 = \times$ . The first rule initializes the infinite path by using the symbol  $\triangleright$  to indicate that there is no element to the left, marks the identity of the group as part of the path by using  $\times$  and sets the next element in the direction  $g_1$ . The second and third rules mark the left and right neighbors respectively and move to the next position. Rule 4 deals with the case of reaching a position already marked and going back. Rule 5 and 6 search the next available direction which potentially admits an infinite path and backtrack if every position has already been searched. Rule 6 lacks a case where  $i = 2k$  on purpose because such a state is never reached as the group is infinite.

Next we construct  $\mathcal{M}_{\text{VISIT}}$  that visits all elements of every ball  $B_n$  in  $G$ . It suffices to construct it as a multiple head  $G$ -machine with three layers as follows: The first layer runs  $\mathcal{M}_{\text{PATH}}$ . The second layer makes use of the path defined by  $\mathcal{M}_{\text{PATH}}$  to simulate a counter which has value  $n \in \mathbb{N}$  – any one-sided Turing machine can be simulated in the path by identifying the instructions  $L, R$  with the first and third coordinates of  $\Sigma$ . The third layer runs another copy of  $\mathcal{M}_{\text{PATH}}$  which is allowed only to run over words of length  $n$ . This is achieved by using the counter in second layer to measure the length of the path visited by the third layer and restrict it to be less than  $n$ . Each time the whole ball  $B_n$  is visited (that is,  $((\triangleright, \times, g_{2k}), B)$  is reached in the third layer) then the counter in the second layer increments  $n \leftarrow n+1$  and the third layer starts anew. If at a given time the first layer which constructs the one-sided path backtracks until reaching a cell used by the counter in the second layer then the second and third layers are erased and restart. As the group is infinite, then by choosing an adequate number of computation steps the path generated by  $\mathcal{M}_{\text{PATH}}$  in the first layer is arbitrarily long, and thus, the head of the third tape is able to visit every element of  $B_n$  for arbitrarily big  $n$ .

Finally, we describe the functioning of  $\mathcal{M}$ . The second and third layers run independently as described above. Whenever the machine  $\mathcal{M}_{\text{VISIT}}$  arrives at a position where the first layer is not marked by  $\sqcup$ , the head at the fourth layer follows the path  $w$  marked from  $1_G$  by the first layer of  $\mathcal{M}_{\text{VISIT}}$  and writes it along with the symbol  $a$  in the sixth layer. That is, the coding  $(w, a)$  is added to the simulated reading tape of the fifth layer, then it marks position  $w$  as already visited (visited symbols count as reading  $\sqcup$  in the first layer) and returns to  $1_G$ . The fifth layer consists of a reading tape where the input written by the fourth layer is stored and a working tape which simulates  $T$  over that input. If a given time the fourth layer extends the pattern coding written in the reading tape of the fifth layer, then the working tape of the fifth layer erases everything and begins anew. If in any moment the end of the simulated path created by the second layer backtracks until reaching a cell used by the written portion in the

simulated tape in the fifth layer, then the content of all the other tapes is erased and they start anew.  $\mathcal{M}$  accepts if and only if the simulated machine  $T$  in the fifth layer reaches an accepting state.

As  $\mathcal{M}_{\text{PATH}}$  is able to construct arbitrarily long one-sided and non-intersecting paths, there is a finite number of computation steps such that  $\mathcal{M}_{\text{VISIT}}$  will visit all of  $\text{supp}(P)$  and thus the fourth layer will write a consistent pattern coding  $c = (w_i, a_i)_{1 \leq i \leq n}$  which is accepted in the fifth layer if and only if  $P = \Pi(c) \in \mathcal{F}$ . By considering a path which has length at least two times the running time of all the other algorithms combined we deduce that if  $P \in \mathcal{F}$  then  $\mathcal{M}$  accepts. Conversely, if  $P \notin \mathcal{F}$ , as  $\mathcal{F}$  is maximal, then  $P \in \mathcal{L}(X)$ . As every  $Q \sqsubset P$  is also in  $\mathcal{L}(X)$  then in any step of the computation the machine  $T$  in the fifth layer can not accept as it works on a coding of a pattern  $Q \sqsubset P$ . And this would mean that  $P$  belongs to  $\mathcal{F}$ . These two statements imply that  $\mathcal{M}$  recognizes  $\mathcal{F}$ .  $\square$

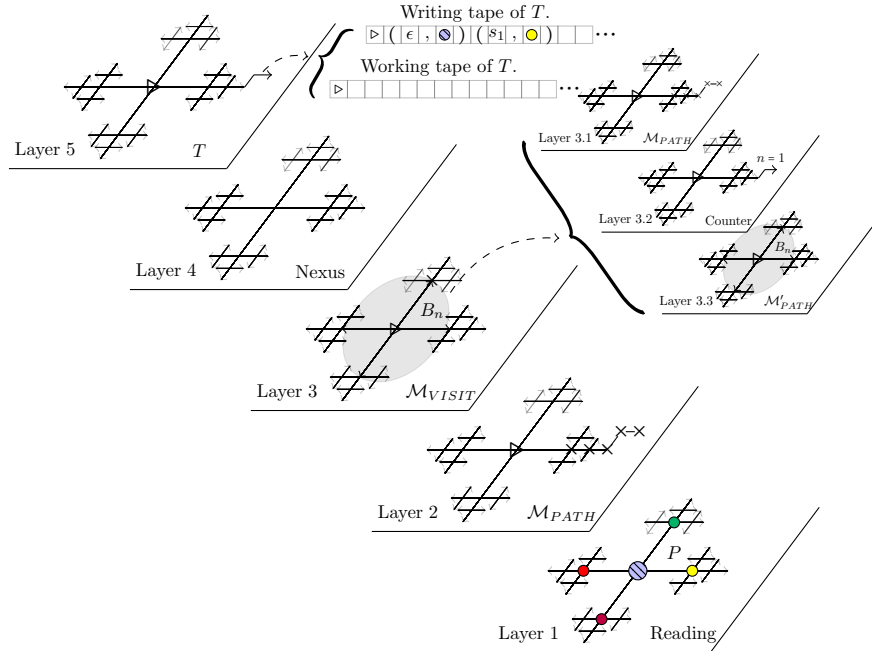


Figure 3: Construction of the machine  $\mathcal{M}$  as a multiple head  $G$ -machine.

We have shown that every  $\mathbb{Z}$ -effective subshift is  $G$ -effective, now we show that the class of  $G$ -effective subshifts is strictly bigger for finitely generated groups with undecidable word problem. Recall that  $X_{\leq 1}$  is not  $\mathbb{Z}$ -effective in this case (Proposition 2.3)

**Proposition 2.7.**  $X_{\leq 1}$  is  $G$ -effective for every finitely generated group  $G$ .

*Proof.* A  $G$ -machine can be constructed that accepts all forbidden patterns. By using  $\mathcal{M}_{\text{VISIT}}$  as in the last theorem we visit all elements in  $B_n$  for  $n \in \mathbb{N}$ . If one symbol 1 is found, erase it and move to a warning state. If another symbol 1 is found, accept the pattern as forbidden.  $\square$

### 2.3 Groups with decidable word problem

**Theorem 2.8.** Let  $G$  be a finitely generated group with decidable word problem and  $X \subset \mathcal{A}^G$  a  $G$ -effective subshift. Then  $X$  is a  $\mathbb{Z}$ -effective subshift.

*Proof.* Let  $G$  be generated by the finite set  $S$ . As the word problem is decidable Proposition 2.1 implies that there is a Turing machine that decides whether a pattern coding is consistent. Thus it suffices to show there is another Turing machine which recognizes all consistent pattern codings of forbidden patterns.

Let  $\mathcal{M}$  the  $G$ -machine which recognizes  $\mathcal{F} \subseteq \mathcal{A}_G^*$  such that  $X = X_{\mathcal{F}}$ . Using the Turing machine which decides the word problem over all words of length  $n \in \mathbb{N}$  it is possible to codify the finite graph  $B_n \subset \Gamma(G, S)$  in the tape of a Turing machine. It is also possible to simulate the functioning of  $\mathcal{M}$  on  $B_n$  by codifying the vertex where the head is located and simulating the moves in  $S$  by changing the position of the head accordingly.

Now let be  $c = (w_i, a_i)_{1 \leq i \leq n}$  a consistent pattern coding and consider a variable  $N \in \mathbb{N}$  and assign initially  $N \leftarrow \max_{1 \leq i \leq n} |w_i|$ . As the pattern coding is consistent it is possible to simulate  $B_N$  and write  $\Pi(c)$  over it. If the simulation of  $\mathcal{M}$  over this initial pattern in  $B_N$  accepts it means the pattern lives in  $\mathcal{F}$  and thus the algorithm accepts. If the simulation eventually needs to move outside  $B_N$  then it restarts the same procedure with  $N \leftarrow N + 1$ .

Using these two machines we can construct one that accepts only if either the pattern coding is inconsistent or if it belongs to  $\mathcal{F}$ , thus  $X$  is  $\mathbb{Z}$ -effective.  $\square$

**Remark** The previous result shows that in terms of effectiveness,  $G$ -effective subshifts are the  $G$ -subshifts defined by a set of forbidden patterns that can be recognized – in the sense of pattern codings – by a Turing machine which has access to an oracle for  $WP(G)$ . This could have been provided as the initial definition but it's not as natural as using  $G$ -machines because it is needed to code the patterns instead of just writing them on the tape. Nevertheless, this characterization is extremely useful in order to show that a  $G$ -subshift is  $G$ -effective.

**Remark** This last remark allows us to say that the  $G$ -recognizable condition in the definition of  $G$ -effectiveness can be replaced by  $G$ -decidable. Indeed, by replacing a pattern  $P \in A^S$  by all patterns with support  $B_n$ ,  $S \subset B_n$ , which contain  $P$ , one can transform an enumeration of patterns by an increasing enumeration that defines the same  $G$ -subshift. Nevertheless, this new  $G$ -decidable set is not necessarily maximal.

In view of this result, if the word problem of  $G$  is decidable, we will speak simply of effective subshifts without making reference to which definition we are using and thus recovering the original notation for effectiveness in  $\mathbb{Z}$ .

## 2.4 Relation between effectiveness and other classes

It has been shown that  $\mathbb{Z}$ -effective subshifts defined over a group  $G$  are always  $G$ -effective, and that the reciprocal holds only for groups with decidable word problem. The inclusion is strict as  $X_{\leq 1}$  provides an example of a  $G$ -effective subshift which is not  $\mathbb{Z}$ -effective. One might wonder about the relation between other classes such as the class of  $G$ -subshifts of finite type or the class of sofic  $G$ -subshifts. In order to study this it is necessary first to state some properties of effectiveness.

**Proposition 2.9.** *The classes of  $\mathbb{Z}$ -effective and  $G$ -effective subshifts are closed under factors.*

*Proof.* Let  $X \subset (\mathcal{A}_X)^G$  be a  $\mathbb{Z}$ -effective subshift and  $\phi : X \rightarrow Y$  be a factor code. There is then a local function  $\Phi : (\mathcal{A}_X)^F \rightarrow \mathcal{A}_Y$  for  $F \subset G$  a finite set such that  $\phi = \Phi_\infty$ . Consider a pattern coding  $c = (w_i, a_i)_{1 \leq i \leq n}$  with  $a_i \in \mathcal{A}_Y$ . W.l.o.g we can assume that each symbol  $a \in \mathcal{A}_Y$  has a preimage under  $\Phi$  and thus we can associate to each  $(w_i, a_i)$  a new consistent pattern coding  $(w_i f, b_{i,f})_{f \in F}$  such that  $\prod((w_i f, b_{i,f})_{f \in F}) \in \Phi^{-1}(a_i)$ . As  $F$  is finite and  $\forall a \in \mathcal{A}_Y, |\Phi^{-1}(a)| < \infty$  this can be done algorithmically. By doing this for every pair  $(w_i, a_i)$  we obtain a pattern coding of a preimage of  $c$ . Now construct algorithmically each of these possible pattern codings – they are finite – and run on each one of them the algorithm which recognizes either an inconsistent pattern coding or one that codifies an elements of  $\mathcal{F}_X \subset (\mathcal{A}_X)_G^*$  such that  $X = X_{\mathcal{F}_X}$ . Accept if and only if the previous algorithm accepts in every case.

The set of patterns associated to consistent pattern codings where this algorithm accepts defines a set  $\mathcal{F}_Y$  and we claim  $Y = Y_{\mathcal{F}_Y}$ . If  $y \in Y$  there is  $x \in X$  with  $\phi(x) = y$ . Let  $P \subset y$ . then  $x|_{\text{supp}(P)F} \notin \mathcal{F}_X$  is eventually codified and the algorithm cannot accept, thus  $P \notin \mathcal{F}_Y$  and  $y \in Y_{\mathcal{F}_Y}$ . Conversely, if  $y \in Y_{\mathcal{F}_Y}$  that means that  $\forall n \in \mathbb{N} y|_{B_n} \notin \mathcal{F}_Y$  so there exists an associated preimage  $Q_n \notin \mathcal{F}_X$ . Consider a sequence  $(x_n)_{n \in \mathbb{N}}$  of  $x \in (\mathcal{A}_X)^G$  such that  $(x_n)|_{B_n F} = Q_n$ . By compactness there is a subsequence which converges to  $x \in (\mathcal{A}_X)^G$ . As for each  $n \in \mathbb{N} x|_{B_n F} \notin \mathcal{F}_X$  and  $B_n \nearrow G$  we have that  $x \in X_{\mathcal{F}_X} = X$  and thus as  $\phi(x_n)|_{B_n} = (y_n)|_{B_n}$  we have by continuity of  $\phi$  that  $y = \phi(x) \in Y$ .

In the case of a  $G$ -effective subshift the proof is the same as it just adds the power of an oracle for the word problem.  $\square$

This result also shows that these classes are invariant under conjugacy. Next we show that the class of sofic  $G$ -subshifts is contained in the class of  $G$ -effective subshifts. As effectiveness is closed under factors, it suffices to prove the inclusion for  $G$ -subshifts of finite type.

**Theorem 2.10.** *Let  $X$  be a sofic  $G$ -subshift, then  $X$  is  $G$ -effective. Moreover, if  $G$  is recursively presented then  $X$  is  $\mathbb{Z}$ -effective.*



*Proof.* As  $G$ -effectiveness can be seen as  $\mathbb{Z}$ -effectiveness where the Turing machine has access to an oracle for  $WP(G)$ , it suffices to do the proof in the latter case assuming just that  $G$  is recursively presented. In virtue of Proposition 2.9 we can assume  $X$  is a  $G$ -SFT and thus there is a finite set  $\mathcal{F} \subseteq (\mathcal{A}_X)_G^*$  such that  $X = X_{\mathcal{F}}$ . Associate with each of these  $Q \in \mathcal{F}$  a consistent pattern coding  $c_Q$  such that  $Q = \Pi(c_Q)$  and consider the algorithm that does the following given a pattern coding  $c = (w_i, a_i)_{1 \leq i \leq n}$ : First consider a variable  $N \leftarrow \max_{1 \leq i \leq n} |w_i|$  and simulate the algorithm which recognizes  $WP(G)$  up to  $N$  steps for each pair of words  $w_1, w_2$  in  $\bigcup_{1 \leq j \leq N} (S \cup S^{-1})^j$  – that is, recognize up to  $N$  steps if  $w_1 w_2^{-1} =_G 1_G$ . If this algorithm returns that  $w_1 =_G w_2$  in less than  $N$  steps then write  $w_1 \sim_N w_2$  and close  $\sim_N$  symmetrically and transitively so that it becomes an equivalence relation. Consider  $\Gamma_N := \bigcup_{1 \leq j \leq N} (S \cup S^{-1})^j / \sim_N$  and assign to each equivalence class of  $w_i$  in  $\Gamma_N$  the symbol  $a_i$ . If two different symbols are assigned to a same equivalence class the pattern is inconsistent and the algorithm accepts. If not, for each  $v \in \bigcup_{1 \leq j \leq N} (S \cup S^{-1})^j$  and  $c_Q = (u_i, b_i)_{1 \leq i \leq m}$  for  $Q \in \mathcal{F}$  consider  $vc_Q = (vu_i, b_i)_{1 \leq i \leq m}$  and check if every word  $vu_i$  is contained in an equivalence class of  $\Gamma_N$ . If it happens to be the case, search if for every  $i$  the equivalence class of  $vu_i$  has the symbol  $b_i$  assigned. If it is the case the algorithm accepts, if not, it assigns  $N \leftarrow N + 1$  and repeats the procedure.

This algorithm recognizes all inconsistent pattern codings and all consistent patterns which contain a  $Q \in \mathcal{F}$ . Indeed, as  $\text{supp}(\Pi(c)) \subset B_{\max_{1 \leq i \leq n} |w_i|}$  and  $WP(G)$  is recognizable there is a  $M \in \mathbb{N}$  such that all the identifications amongst elements of length at most  $\max_{1 \leq i \leq n} |w_i|$  are found and thus if there is  $Q \in \mathcal{F}$  such that  $Q \subset \Pi(c)$ , then it is found before step  $M + 1$ . Conversely, if  $\Pi(c)$  does not contain any pattern in  $\mathcal{F}$  the algorithm will never accept.  $\square$

The previous theorem lets us construct the following diagram between these classes.

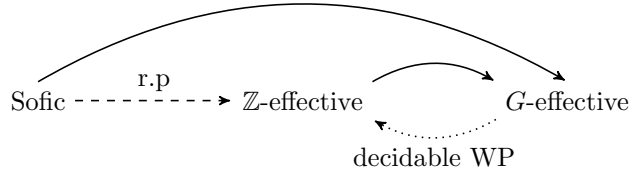


Figure 4: Inclusion relations between different classes of  $G$ -subshifts for a finitely generated group  $G$ . Inclusion represented by a dashed arrow only holds for recursively presented groups, and inclusion represented by a dotted arrow only for groups with decidable word problem.

### 3 Groups with $G$ -effective subshifts which are not sofic.

An interesting question with respect to Figure 4 is if the inclusion of sofic  $G$ -subshifts with respect to  $G$ -effective subshifts is always strict. Proposition 2.3 together with Theorem 2.10 allows to give a partial answer to that question.

**Theorem 3.1.** *For every finitely generated and recursively presented group  $G$  with undecidable word problem the  $G$ -subshift  $X_{\leq 1} \subset \{0, 1\}^G$  is not sofic.*

In [10], the authors define the *special symbol property* for a group  $G$  as the property of the one-or-less  $G$ -subshift  $X_{\leq 1}$  being sofic. They prove that groups satisfying special symbol property are finitely generated (Proposition 1.6) and give several examples of groups with this property (Section 4). Theorem 3.1 provides a wide class of finitely generated groups that do not enjoy the special symbol property: those which are recursively presented and with undecidable word problem.

Sofic  $\mathbb{Z}$ -subshifts admit a characterization with finite automata [18]. For higher dimensional sofic  $\mathbb{Z}^d$ -subshifts no characterization is known, but some examples of effective subshifts which are not sofic are known. In this section we do a quick review of a famous example for  $\mathbb{Z}^2$  and generalize the essential part of the proof in order to construct an example of  $G$ -effective but not sofic  $G$ -subshift for infinite amenable groups. We also construct another example to give the same result for groups having more than two ends.

#### 3.1 The $\mathbb{Z}^d$ case

Let  $\mathcal{A} = \{\square, \blacksquare, \color{red}\blacksquare\}$  and consider the following set of forbidden patterns:

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

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  $1 \leq i \leq |w|$ .

Denote by  $X_{\text{mirror}}$  the subshift  $X_{F_{\text{mirror}}} \subset \mathcal{A}^{\mathbb{Z}^2}$ . This  $\mathbb{Z}^2$ -subshift, called the *mirror subshift*, contains the  $\mathbb{Z}^2$ -fullshift  $\{\square, \blacksquare\}^{\mathbb{Z}^2}$  as a subsystem, but also all configurations that respect the following conditions: a symbol  $\color{red}\blacksquare$  forces all symbols in the same column to be also  $\color{red}\blacksquare$  symbols; there is at most one column of  $\color{red}\blacksquare$  symbols; if a symbol  $\color{red}\blacksquare$  is present on a row, then  $\square$  and  $\blacksquare$  symbols of this row are arranged symmetrically with respect to the  $\color{red}\blacksquare$  symbol.

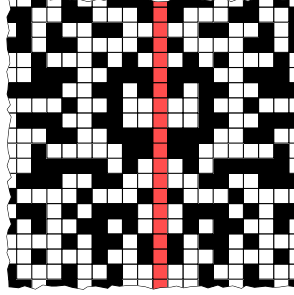


Figure 5: One configuration in the two-dimensional mirror subshift  $X_{\text{mirror}}$ .

The column of  $\blacksquare$ , 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 effective – we say plainly effective as  $WP(\mathbb{Z}^2)$  is decidable – since the set of forbidden patterns  $F_{\text{mirror}}$  can be effectively enumerated, but one can prove it is not sofic by a direct combinatorial argument.

**Proposition 3.2.** *The mirror subshift  $X_{\text{mirror}} \subset \mathcal{A}^{\mathbb{Z}^2}$  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  $X \subset \mathcal{B}^{\mathbb{Z}^2}$  on some finite alphabet  $\mathcal{B}$  and a 1-block factor code  $\phi : X \rightarrow X_{\text{mirror}}$ .

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

Consider now all configurations  $x \in X_{\text{mirror}}$  in which a mirror appears at the origin, that is to say  $x_{(0,j)} = \blacksquare$  for all  $j \in \mathbb{Z}$ . For  $n$  large enough one has  $|\mathcal{B}|^{4(2n+2)} < 2^{(2n+1)^2}$ , consequently there exist two distinct patterns  $P_1$  and  $P_2$  with support  $\Lambda_n$  that appear respectively in configurations  $y_1$  and  $y_2$  of  $X_{\text{mirror}}$  – 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  $X_{\text{mirror}}$  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  $\tilde{y} \in \mathcal{A}^{\mathbb{Z}^2}$  defined by

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

in other terms  $\tilde{y}$  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  $\tilde{y}$  a mirror appears at the origin, but since  $P_1$  and  $P_2$  have been chosen distinct  $\tilde{y} \notin X_{\text{mirror}}$ . On the

other hand the configuration  $\tilde{x} \in \mathcal{B}^{\mathbb{Z}^2}$  defined by

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

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

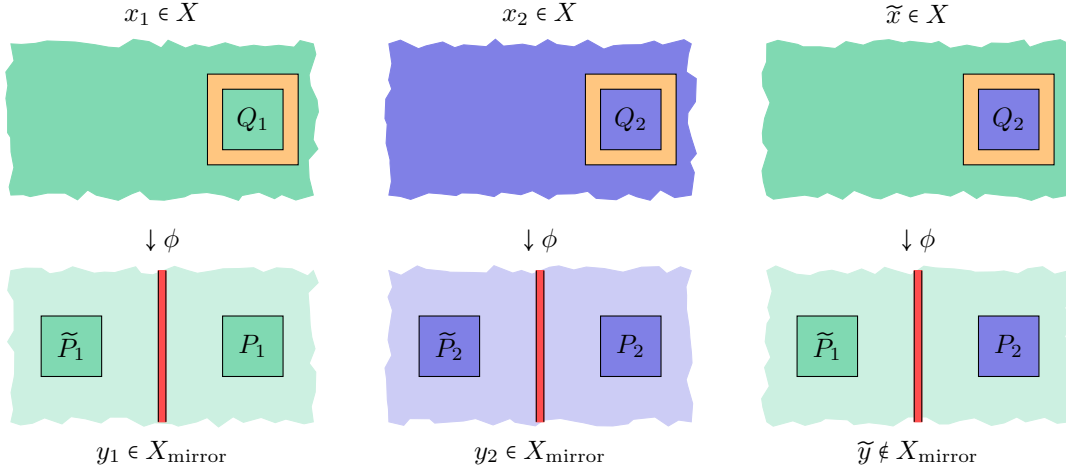


Figure 6: Two configurations  $y_1$  and  $y_2$  in the mirror subshift  $X_{\text{mirror}}$  with a mirror at the origin, and that differ on  $\Lambda_n + (n^2, 0)$ , but whose pre-images in the nearest neighbor  $\mathbb{Z}^2$ -SFT extension  $X$  are the same on  $\partial\Lambda_{n+1}$ . If it were so, one could construct a configuration  $\tilde{y}$  – 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  $X_{\text{mirror}}$ . This proves  $X_{\text{mirror}}$  is not sofic.

□

**Remark** 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  $x \in \mathcal{A}^{\mathbb{Z}^d}$  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 3.2 can be generalized to any dimension.

The key ingredients in the proof of Proposition 3.2, with the aim of generalizing the result to a bigger class of finitely generated groups, are the following:

1. A  $G$ -effective subshift  $X$  with highly non-local conditions, that is to say there exist arbitrarily distant elements in  $G$  that are forced to share the same symbol;

2. The existence of an increasing sequence of finite sets  $F_n$  in  $G$  – in  $\mathbb{Z}^2$  they are the square balls  $\Lambda_n$  – whose border  $\partial F_n$  grows slower than the sets themselves.

### 3.2 Amenable groups

In this section we prove that any infinite and finitely generated amenable group admits  $G$ -effective subshifts that are not sofic (Theorem 3.4). A group  $G$  is called *amenable* if there exists a left-invariant finitely additive probability measure  $\mu : \mathcal{P}(G) \rightarrow [0, 1]$  on  $G$ . The amenability of a group has many equivalent definitions – many of which can be found in [7]. From a combinatorial point of view the Følner condition states that a group is amenable if and only if it admits a Følner net, that is, a net  $(F_\alpha)_{\alpha \in J}$  of non-empty finite sets  $F_\alpha \subset G$  such that  $\forall g \in G$ :

$$\lim_{\alpha} \frac{|F_\alpha \setminus F_\alpha g|}{|F_\alpha|} = 0.$$

In the case of a finitely generated group –as they are countable– the net can be just taken to be a sequence and thus amenability can be shown to be equivalent to the fact that  $\inf_{F \subset G, |F| < \infty} |\partial F|/|F| = 0$ . There is a sequence of sets such that their boundary grows slower than themselves.

In order to generalize the proof to this setting the mirror subshift defined in Section 3.1 is the natural candidate, but its definition requires the existence of a torsion free element in the group. One way to get over this constraint would be to consider the *symmetric subshift*  $X_{sym} \subset \mathcal{A}^G$  defined on alphabet  $\mathcal{A} = \{0, 1, \boxtimes\}$  as the set of  $x \in \mathcal{A}^G$  such that  $|\{g \in G : x_g = \boxtimes\}| \leq 1$  and if for  $\bar{g} \in G$   $x_{\bar{g}} = \boxtimes$  then  $\forall h \in G$  we have  $x_{\bar{g}h} = x_{\bar{g}h^{-1}}$ . That is, the mirror  $\boxtimes$  is now located in only one position and it forces elements to carry the same symbol as their inverses with respect to the position of the mirror. The problem with this construction is that the inverse of a ball  $gB_n$  is not necessarily a ball for the same set of generators, and can behave badly. An analogous proof to the one of Theorem 3.4 can be done using  $X_{sym}$  in the case that  $G$  satisfies the following condition:  $\exists \alpha \in [0, 1)$  such that  $\forall F \subset G$  with  $F$  finite,  $1_G \in F$  and  $F = F^{-1}$  then  $\exists g \in G : |gFg \cap F| \leq \alpha|F|$ . In other words, that there is an element  $g \in G$  such that the inverse of a big enough ball  $B$  centered in  $g$  does not intersect itself in more than a bounded constant proportion.

We get rid of this additional condition by considering instead a new construction which enforces two sequences of arbitrarily disjoint balls  $(B_n)$  to mimic each other in pairs.

**Definition** Let  $G$  be an infinite group generated by a finite set  $S$  and let  $\mathcal{G} = (g_i)_{i \in \mathbb{N}} \subset G$  and  $\mathcal{H} = (h_i)_{i \in \mathbb{N}} \subset G$  be two sequences such that:

- The sequences of sets  $(g_i B_i)_{i \in \mathbb{N}}$  and  $(h_i B_i)_{i \in \mathbb{N}}$  are pairwise disjoint

$$\forall i \neq j, h_i B_i \cap h_j B_j = g_i B_i \cap g_j B_j = h_i B_i \cap g_j B_j = h_i B_i \cap g_i B_i = \emptyset.$$

- $\forall i \in \mathbb{N}, 1_G \notin g_i B_i \cup h_i B_i$ .

We define the *ball mimic subshift*  $X_B(\mathcal{G}, \mathcal{H}) \subset \{0, 1, \boxtimes\}^G$  as the  $G$ -subshift such that in every configuration  $x \in X_B(\mathcal{G}, \mathcal{H})$  the symbol  $\boxtimes$  appears at most once, and if for  $\bar{g} \in G$   $x_{\bar{g}} = \boxtimes$  then  $\forall i \in \mathbb{N} \sigma_{(\bar{g}g_i)^{-1}}(x)|_{B_i} = \sigma_{(\bar{g}h_i)^{-1}}(x)|_{B_i}$ . Formally  $X_B(\mathcal{G}, \mathcal{H}) := X_{\mathcal{F}}$  where  $\mathcal{F} = \{\Pi^g, \Upsilon^{i,s} | g \in G \setminus \{1_G\}, i \in \mathbb{N}, s \in B_i\}$  where  $\text{supp}(\Pi^g) = \{1_G, g\}$ ,  $\text{supp}(\Upsilon^{i,s}) = \{1_G, g_i s, h_i s\}$ ,  $\Pi_{1_G}^g = \Pi_g^g = \Upsilon_{1_G}^{i,s} = \boxtimes$ ,  $\Upsilon_{g_i s}^{i,s} \neq \Upsilon_{h_i s}^{i,s}$ .

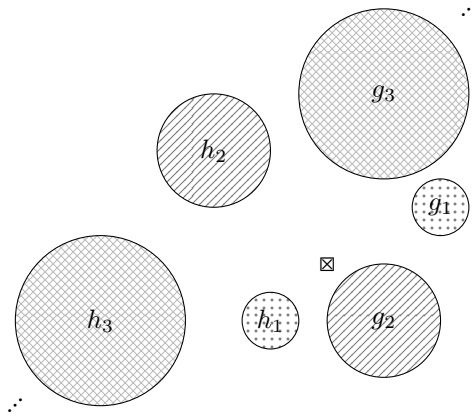


Figure 7: The ball mimic subshift given by two sequences  $(g_i)_{i \in \mathbb{N}}$  and  $(h_i)_{i \in \mathbb{N}}$ . If the symbol  $\boxtimes$  appears in position  $\bar{g}$ , then  $\forall i \in \mathbb{N}$ , the patterns  $x_{\bar{g}g_i B_i}$  and  $x_{\bar{g}h_i B_i}$  are identical.

**Proposition 3.3.** *Let  $G$  be an infinite group and  $S \subset G$  a finite set of generators. Then there are sequences  $\bar{\mathcal{G}}, \bar{\mathcal{H}}$  such that  $X_B(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  is  $G$ -effective.*

*Proof.* We construct a  $G$ -machine which recognizes exactly the forbidden patterns  $\Pi^g$  and  $\Upsilon^{i,s}$  given in the definition. In the previous section we showed that  $G$ -machines are capable of running Turing machines in a 1-sided tape simulated by  $\mathcal{M}_{PATH}$  and are also able via  $\mathcal{M}_{VISIT}$  to visit every element in  $B_n$  for a given set of generators  $S$ . In that sense, we can write loosely the functioning of the  $G$ -machine as described by Algorithm 1.

In this algorithm we explicitly did two abuses of notation to simplify the presentation. The first is that whenever  $\mathcal{M}_{VISIT}$  is used to visit all the elements of a ball  $B_n$  and an instruction is carried over them, we assume we carry the instruction only once (and thus, we never write the symbol  $\times$  in the auxiliary pattern  $Q$ ). The second abuse of notation is that whenever we check the condition  $P_{h_i f} \neq Q_f$  we assume that neither of the two symbols is  $\sqcup$ . If  $P_{h_i f}$  or

```

Data:  $P \in \{0, 1, \boxtimes\}_G^*$ 
1  $\bar{g} \leftarrow 1_G, n \leftarrow 1;$ 
2 while True do
3   Visit all  $h \in B_n$ . If  $P_h = \boxtimes$  then  $\bar{g} \leftarrow h$ , Break;
4    $n \leftarrow n + 1;$ 
5 end
6  $i \leftarrow 1, P_{\bar{g}} \leftarrow \times;$ 
7 while True do
8   Visit all  $h \in \bar{g}B_i$ . If  $P_h = \boxtimes$  Accept;
9    $Q \leftarrow \sqcup^{B_i}, m \leftarrow 1;$ 
10  while True do
11    Visit all  $h \in \bar{g}B_m$ . if  $P_h \neq \times$  then
12       $g_i \leftarrow h;$ 
13      Visit all  $f \in g_iB_i$ . If  $\forall f \in g_iB_i P_f \neq \times$  Break;
14    end
15     $m \leftarrow m + 1;$ 
16  end
17  Visit all  $f \in B_i$ .  $Q_f \leftarrow P_{g_i f}, P_{g_i f} \leftarrow \times;$ 
18   $m \leftarrow 1;$ 
19  while True do
20    Visit all  $h \in \bar{g}B_m$ . if  $P_h \neq \times$  then
21       $h_i \leftarrow h;$ 
22      Visit all  $f \in h_iB_i$ . If  $\forall f \in h_iB_i P_f \neq \times$  Break;
23    end
24     $m \leftarrow m + 1;$ 
25  end
26  Visit all  $f \in B_i$ . if  $P_{h_i f} \neq Q_f$  then
27    Accept;
28  else
29     $P_{h_i f} \leftarrow \times;$ 
30  end
31   $i \leftarrow i + 1;$ 
32 end

```

**Algorithm 1:** Recognizing translations of  $\mathcal{F}$

$Q_f = P_{g_i f}$  are  $\sqcup$  that means that the initial pattern did not contain that position in its support and thus  $P_{h_i f} \neq Q_f$  is assumed by default to be False.

The loop in line 2 backtracks over  $G$  in order to find a  $\boxtimes$ , if it is found, its position is stored in the variable  $\bar{g}$ . In the next loop, line 8 continues to backtrack over the group to find  $\boxtimes$ , if it is found the algorithm accepts, thus identifying all translations of patterns of the form  $\Pi^g$ . The marking of  $\bar{g}$  with a  $\times$  ensures that  $1_G$  does not belong to the sequence of balls. Lines 10 and 19 find sequences  $g_i, h_i$  satisfying the requirement that  $(g_i B_i)_{i \in \mathbb{N}}$  and  $(h_i B_i)_{i \in \mathbb{N}}$  are pairwise disjoint. Finally, line 27 accepts if  $g_i B_i \neq h_i B_i$  and thus it identifies all patterns  $\Upsilon^{i,s}$  as above. Also note that the loop in line 7 behaves exactly the same after the element  $\bar{g}$  is found. Thus the sequences  $\bar{\mathcal{G}} = (\bar{g}^{-1} g_i)_{i \in \mathbb{N}}, \bar{\mathcal{H}} = (\bar{g}^{-1} h_i)_{i \in \mathbb{N}}$  are well defined, satisfy the requirements and in consequence  $X_B(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  is  $G$ -effective.  $\square$

**Theorem 3.4.** *Let  $G$  be an infinite and finitely generated amenable group. Then for all sequences  $\mathcal{G}, \mathcal{H}$  as described above the ball mimic shift  $X_B(\mathcal{G}, \mathcal{H})$  is not sofic.*

*Proof.* Consider a finite set  $S \subset G$  which generates  $G$ .

First note that for any finite set  $F$  then  $\partial F = \bigcup_{s \in S} F \setminus Fs$ , indeed, if  $g \in \partial F$  then there exists  $s \in S$  such that  $gs \notin F$  and thus  $g \in F \setminus Fs$ , conversely, if  $g \in \bigcup_{s \in S} F \setminus Fs$  there is  $s \in S$  such that  $g \in F \setminus Fs$  and thus  $gs \notin F$  but  $g \in F$ .

As explained in [7] one of the characterizations of  $G$  being amenable, is that for every finite set  $K$  and  $\varepsilon > 0$  there exists a finite set  $F$  such that

$$\forall k \in K, \frac{|F \setminus Fk|}{|F|} < \varepsilon$$

Suppose  $X_B(\mathcal{G}, \mathcal{H})$  is sofic, then there exists an  $S$ -nearest neighbor  $G$ -SFT extension  $X \subset \mathcal{B}^G$  and a 1-block factor code  $\phi : X \rightarrow X_B(\mathcal{G}, \mathcal{H})$ .

By choosing  $K = S$  and  $\varepsilon = \frac{\log(2)}{|S| \log(|\mathcal{B}|)}$  we obtain that there is  $F$  such that:

$$\frac{|\partial F|}{|F|} = \frac{|\bigcup_{s \in S} F \setminus Fs|}{|F|} \leq \sum_{s \in S} \frac{|F \setminus Fs|}{|F|} < |S| \frac{\log(2)}{|S| \log(|\mathcal{B}|)} = \frac{\log(2)}{\log(|\mathcal{B}|)}$$

Note that this property is invariant by translation, that is, if  $F$  satisfies this property, then  $gF$  also does for each  $g \in G$ . By choosing a large enough  $n \in \mathbb{N}$  such that  $F \subset B_n$  and  $1_G \notin g_n B_n$  then  $g_n F \subset g_n B_n$  and  $1_G \notin g_n F$ .

Putting everything together, we can find a set  $F$  such that  $|\mathcal{B}|^{|\partial F|} < 2^{|F|}$  and with the properties that  $1_G \notin F$ ,  $\exists n \in \mathbb{N}$  such that  $F \subset g_n B_n$  and  $F \cap h_n B_n = \emptyset$ .

Consider the set of patterns  $t : 1_G \cup F \rightarrow \{\boxtimes, 0, 1\}$  such that  $t_{1_G} = \boxtimes$  and  $t(F) \in \{0, 1\}^F$  and note that there are exactly  $2^{|F|}$  patterns like that. As  $F \subset g_n B_n$  then each cylinder  $[t]_{1_G}$  is non-empty in  $X_B(\mathcal{G}, \mathcal{H})$ . Consider for each  $t$  a configuration  $y^t \in [t]_{1_G} \cap X_B(\mathcal{G}, \mathcal{H})$  and  $x^t \in X \subset \mathcal{B}^G$  with  $\phi(x^t) = y^t$ . As  $|\mathcal{B}|^{|\partial F|} < 2^{|F|}$  by pigeonhole principle there are  $x^{t_1} \neq x^{t_2}$  such that  $x^{t_1}|_{\partial F} = x^{t_2}|_{\partial F}$  and  $y^{t_1} = \phi(x^{t_1}) \neq \phi(x^{t_2}) = y^{t_2}$ . As  $X$  is a nearest neighbor  $G$ -SFT we can construct  $\tilde{x} \in X$  such that  $\tilde{x}|_F = x^{t_1}|_F$  and  $\tilde{x}|_{G \setminus F} = x^{t_2}|_{G \setminus F}$ . As  $\phi$  is a 1-block



code we get  $\phi(\tilde{x})|_F = y^{t_1}|_F$  and  $\phi(\tilde{x})|_{G \setminus F} = y^{t_2}|_{G \setminus F}$ . Consider  $\bar{g} \in B_n$  such that  $g_n \bar{g} \in \text{mathring}F$  with  $y_{g_n \bar{g}}^{t_1} \neq y_{g_n \bar{g}}^{t_2}$  then we get that  $\phi(\tilde{x})_{1_G} = \boxtimes$  and

$$(y^{t_1})_{g_n \bar{g}} = \phi(\tilde{x})_{g_n \bar{g}} \neq \phi(\tilde{x})_{h_n \bar{g}} = (y^{t_2})_{h_n \bar{g}} = (y^{t_2})_{g_n \bar{g}}$$

And thus  $\phi(\tilde{x}) \notin X_B(\mathcal{G}, \mathcal{H})$  hence raising a contradiction.  $\square$

We have thus shown that for each infinite, finitely generated and amenable group the class of  $G$ -effective subshifts is strictly larger than the class of sofic  $G$ -subshifts. The class of amenable groups is quite large. It contains all abelian groups – and thus, this theorem implies the previous result for  $\mathbb{Z}^d$  – moreover, it contains all virtually nilpotent groups – that is, groups with polynomial growth – and even all groups with sub-exponential growth such as the Grigorchuk group [13]. There are also some groups with exponential growth which are amenable. For example, it is known that all solvable groups are amenable and thus the infinite lamplighter group  $\mathbb{Z}_2 \wr \mathbb{Z}$  or the meta-abelian Baumslag-Solitar groups  $(BS(1, n))_{n \in \mathbb{N}}$  are all amenable and with exponential growth if  $n \geq 2$ .

A large class of non-amenable groups is given by those that contain a non-abelian free subgroup. In particular a very well known class are the virtually free groups. We use a different technique to show that the same result proven here holds for a class larger than the virtually free groups.

### 3.3 On free groups

In the case of finitely generated free groups it is straightforward to show the existence of effective – the word problem for free groups is decidable –  $G$ -subshifts which are not sofic.

Consider the symmetric subshift  $X_{sym} \subseteq \{0, 1, \boxtimes\}^G$  given by the set of configurations such that  $|\{g \in G \mid x_g = \boxtimes\}| \leq 1$  and if  $x_{\bar{g}} = \boxtimes$  then  $\forall h \in G \ x_{\bar{g}h} = x_{\bar{g}h^{-1}}$ . It is easy to see that  $X_{sym}$  is  $G$ -effective for every finitely generated group  $G$ . Here we show it cannot be sofic.

**Proposition 3.5.** *For every finitely generated free group  $F_k$  where  $k \geq 1$   $X_{sym}$  is not sofic.*

*Proof.* As  $F_k = \langle S \rangle$  where  $S = \{s_1, \dots, s_k\}$  the Cayley graph  $\Gamma(F_k, S)$  is a  $2k$ -regular infinite tree. Thus  $\Gamma(F_k, S) \setminus \{1_G\}$  has  $2k$  infinite connected components. In particular for  $s = s_1$  the set of elements  $(s^n)_{n \in \mathbb{N}}$  and  $(s^{-n})_{n \in \mathbb{N}}$  live in disjoint components  $C_1$  and  $C_2$ . Suppose  $X_{sym}$  is sofic and thus there is a  $S$ -nearest neighbor  $G$ -SFT extension  $X \subset \mathcal{B}^{F_k}$  and a 1-block code  $\phi : X \twoheadrightarrow X_{sym}$ . As  $(s^n)_{n \in \mathbb{N}}$  and  $(s^{-n})_{n \in \mathbb{N}}$  live in disjoint components it is possible to assign any value to  $(y_{s^n})_{n \in \mathbb{N}} \in \{0, 1\}^{\mathbb{N}}$  and still fill the missing coordinates to construct  $y \in X_{sym} \cap [\boxtimes]_{1_G}$ . Moreover as for  $x \in X$ ,  $x_{1_G}$  can take at most  $|\mathcal{B}|$  values, then by pigeonhole principle there must be  $y_1, y_2 \in X_{sym} \cap [\boxtimes]_{1_G}$  such that there is  $n \in \mathbb{N}$  with  $(y_1)_{s^n} \neq (y_2)_{s^n}$  and  $x_1, x_2 \in X$  with  $(x_1)_{1_G} = (x_2)_{1_G}$ ,  $\phi(x_1) = y_1$  and  $\phi(x_2) = y_2$ . As  $X$  is  $S$ -nearest neighbor it is possible to construct  $\tilde{x}$  such

that  $\tilde{x}|_{C_1} = x_1$  and  $\tilde{x}|_{G \setminus C_1} = x_2$ . Consequently as  $\phi$  is 1-block  $\tilde{y} = \phi(\tilde{x})$  satisfies  $\tilde{y}|_{C_1} = y_1$  and  $\tilde{y}|_{G \setminus C_1} = y_2$ . Thus  $\tilde{y}|_{1_G} = \boxtimes$  and

$$\tilde{y}_{s^n} = (y_1)_{s^n} \neq (y_2)_{s^n} = (y_2)_{s^{-n}} = \tilde{y}_{s^{-n}}$$

Consequently  $\tilde{y} \notin X_{sym}$  thus raising a contradiction.  $\square$

As in the amenable case, we rely also in a subshift with highly non-local conditions, but in contrast to that problem, here the main ingredient doesn't rely on the growth of the border of a finite set, in fact, if  $k \geq 2$  then for  $F_k$  the quotient  $|\partial B_n|/|B_n|$  does not go to 0. Instead we rely in the fact that the Cayley graph can be disconnected by removing a finite set. We take advantage of this idea in order to generalize this proof to groups with two or more ends.

### 3.4 Groups having two or more ends

**Definition** Let  $G$  be a group generated by a finite set  $S \subset G$ . The number of ends  $e(G)$  of the group  $G$  is the limit as  $n$  tends to infinity of the number of infinite connected components of  $\Gamma(G, S) \setminus B_n$ .

The number of ends is a quasi-isomorphism invariant and thus it does not depend on the choice of  $S$  and is a group invariant. It is also known that for a finitely generated group  $G$  then  $e(G) \in \{0, 1, 2, \infty\}$ . Stallings theorem about ends of groups [27] gives a constructive characterization of the groups satisfying  $e(G) \geq 2$ . In particular we have  $e(G) = 2$  if and only if  $G$  is infinite and virtually cyclic. If  $e(G) = \infty$  Stallings theorem implies that  $G$  contains a non-abelian free group. It also shows that every virtually free group satisfies  $e(G) \geq 2$ .

**Theorem 3.6.** *Let  $G$  be a finitely generated group where  $e(G) \geq 2$  then there are  $G$ -effective subshifts which are not sofic.*

*Proof.* Let  $S$  be a finite set of generators for  $G$  and  $N \in \mathbb{N}$  such that  $\Gamma(G, S) \setminus B_N$  contains at least two different infinite connected components  $C_1$  and  $C_2$ .

Let  $\mathcal{G} = (g_i)_{i \in \mathbb{N}} \subset C_1$  and  $\mathcal{H} = (h_i)_{i \in \mathbb{N}} \subset C_2$  be sequences of elements with no repeated elements. We define the *mimic subshift*  $X(\mathcal{G}, \mathcal{H}) \subset \{0, 1, \boxtimes\}^G$  that consists of all configurations containing at most one time the symbol  $\boxtimes$  and such that if  $x_g = \boxtimes$  then  $\forall i \in \mathbb{N} x_{gg_i} = x_{gh_i}$ . Formally  $X(\mathcal{G}, \mathcal{H}) = X_{\mathcal{F}}$  where  $\mathcal{F} = \{\Pi^g, \Upsilon^i \mid g \in G \setminus \{1_G\}, i \in \mathbb{N}\}$  where  $\text{supp}(\Pi^g) = \{1_G, g\}$ ,  $\text{supp}(\Upsilon^i) = \{1_G, g_i, h_i\}$ ,  $\Pi_{1_G}^g = \Pi_g^g = \Upsilon_{1_G}^i = \boxtimes$ ,  $\Upsilon_{g_i}^i \neq \Upsilon_{h_i}^i$ .

There exist sequences  $\bar{\mathcal{G}}, \bar{\mathcal{H}}$  such that  $X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  is  $G$ -effective. The algorithm to construct them is loosely same as the one given for showing the  $G$ -effectiveness of  $X_B(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  with the simplification that here we do not need to identify balls around the sequences, but instead we care that they rest inside a given component.

Consider two fixed elements  $g_1 \in C_1$  and  $h_1 \in C_2$ . A  $G$ -machine which recognizes all translation of the forbidden patterns  $\Pi^g$  and  $\Upsilon^i$  is given by the following: it first visits all of  $(B_n)_{n \in \mathbb{N}}$  in search of a symbol  $\boxtimes$ . If it is found, we mark its position  $\bar{g}$  and we alternate two procedures: in the first one we continue

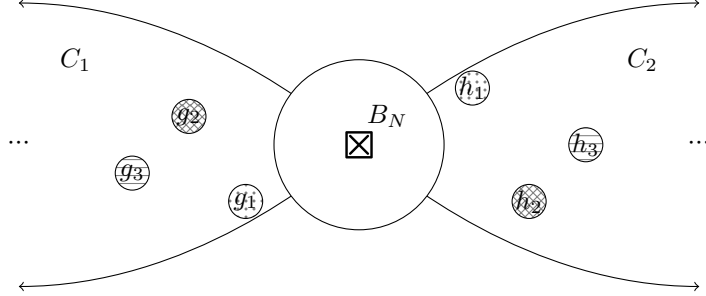


Figure 8: The mimic subshift given by two sequences  $(g_i)_{i \in \mathbb{N}}$  and  $(h_i)_{i \in \mathbb{N}}$ . If the symbol  $\boxtimes$  appears in position  $\bar{g}$ , then  $\forall i \in \mathbb{N}$ , the symbols  $x_{\bar{g}g_i}$  and  $x_{\bar{g}h_i}$  are identical.

visiting  $(B_n)_{n \in \mathbb{N}}$  in search of another  $\boxtimes$ , if it is found the algorithm accepts, thus recognizing all translation of patterns  $\Pi^g$ . In the second procedure we set  $i \leftarrow 1$  and mark  $\bar{g}B_N$  with a special symbol  $\otimes$  and do the following in a loop. First we check  $\bar{g}g_i$  and  $\bar{g}h_i$ . If they are different the algorithm accepts a forbidden pattern, and if not, it marks them with  $\times$ , assigns  $i \leftarrow i + 1$  and visits every element using  $\mathcal{M}_{VISIT}$  starting from those positions with the restriction of not being able to pass by elements marked by  $\otimes$ . Whenever new elements  $g^*$  and  $h^*$  not marked by  $\times$  are found in  $C_1$  and  $C_2$  respectively we set  $g_i := \bar{g}^{-1}g^*$ ,  $h_i := \bar{g}^{-1}h^*$  and iterate the loop.

As the backtracking is not allowed to pass by elements in  $B_n$ , each one of them rests inside their components  $C_1$  and  $C_2$  respectively, and the omission of elements marked by  $\times$  implies that the sequences are of distinct elements.  $\bar{\mathcal{G}} = (g_i)_{i \in \mathbb{N}}$  and  $\bar{\mathcal{H}} = (h_i)_{i \in \mathbb{N}}$  thus satisfy the requirements and the  $G$ -machine described above is able to detect every translation of the patterns  $\Upsilon^i$ , and thus  $X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  is  $G$ -effective.

We argue by contradiction and suppose that  $X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  is also a sofic  $G$ -subshift. W.l.o.g we can choose a  $S$ -nearest neighbor  $G$ -SFT extension  $X \subset \mathcal{B}^G$  given by a 1-block code  $\phi: X \rightarrow X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$ .

As  $G$  is finitely generated then  $B_N$  is finite and there are at most  $|\mathcal{B}|^{|B_N|}$  possible configurations  $t$  for this ball such that  $\phi([t]_{1_G}) \subset [\boxtimes]_{1_G}$ . If  $y \in X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  satisfies that  $y_{1_G} = \boxtimes$  then the symbols  $(y_{g_i})_{i \in \mathbb{N}}$  can be arbitrarily chosen in  $\{0, 1\}^{\mathbb{N}}$  and construct a valid point. By pigeonhole principle there exist  $y_1, y_2 \in [\boxtimes]_{1_G} \cap X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  such that  $\exists i \in \mathbb{N}$   $(y_1)_{g_i} \neq (y_2)_{g_i}$  and  $x_1, x_2 \in X$  satisfying  $\phi(x_1) = y_1$ ,  $\phi(x_2) = y_2$  and  $(x_1)|_{B_N} = (x_2)|_{B_N}$ .

As  $X$  is  $S$ -nearest neighbor the possible configurations on  $C_1$  and  $C_2$  depend exclusively on the configuration of  $B_N$  and thus one can construct  $\tilde{x} \in X$  such that:

$$\tilde{x}_g = \begin{cases} (x_1)_g, & \text{if } g \in C_1 \\ (x_2)_g, & \text{if } g \in G \setminus C_1 \end{cases}$$

Thus  $\tilde{y} = \phi(\tilde{x})$  satisfies that  $\tilde{y}|_{1_g} = \emptyset$ ,  $\tilde{y}|_{C_1} = (y_1)|_{C_1}$  and  $\tilde{y}|_{C_2} = (y_2)|_{C_2}$ . Thus

$$\tilde{y}_{g_i} = (y_1)_{g_i} \neq (y_2)_{g_i} = (y_2)_{h_i} = \tilde{y}_{h_i}$$

And thus  $\tilde{y} \notin X(\bar{\mathcal{G}}, \bar{\mathcal{H}})$  hence giving the desired contradiction. □

Note that Theorem 3.6 implies that every virtually free group admits effective subshifts which are not sofic – we say effective as the word problem is context free and thus decidable for virtually free groups [20].

## Conclusion

We defined a natural notion of effectiveness for  $G$ -subshifts over finitely generated groups, by generalizing classical Turing machines so that they use the group as the tape. The class of effective subshifts it defines extends the classical notion – which we have proved makes only sense for groups with decidable word problem – and contains the class of sofic  $G$ -subshifts. We have also shown that for a finitely generated group  $G$  there exist  $G$ -effective subshifts which are not sofic in the following three cases:

1. recursively presented groups with undecidable word problem,
2. infinite amenable groups,
3. groups which have two or more ends.

Some groups which do not necessarily fall under these categories are counterexamples to the von Neumann conjecture, that is, groups which are not amenable but do not contain  $F_2$  the free group on two generators as a subgroup. An uncountable family of non-isomorphic finitely generated counterexamples is given by the Tarski monster groups found by Ol’shanskii [22], and thus, by cardinality, there are some which are not recursively presented and which are not covered in the scope of our results. A class of finitely presented groups which is non-amenable and has no free subgroups has been constructed in [23]. For all these examples there are no known techniques to prove non soficness – at least to the knowledge of the authors. We end by stating three questions.

## Questions

1. Are there infinite and finitely generated groups  $G$  s.t. the class of  $G$ -effective subshifts matches with the class of sofic  $G$ -subshifts?
2. Are there infinite and finitely generated groups  $G$  s.t. the class of  $\mathbb{Z}$ -effective subshifts matches with the class of sofic  $G$ -subshifts?

3. Is the one-or-less  $G$ -subshift  $X_{\leq 1}$  sofic if and only if  $G$  has decidable word problem?

Note that in the case of recursively presented groups, these three questions can be reformulated. Indeed, a group answering positively the first question would also answer positively the second. In this context the second question is different from the first only for groups with undecidable word problem. In the third question one direction is already proven: soficness implies decidability for recursively presented groups.

## Acknowledgments

The authors would like to thank Yves de Cornulier for interesting discussions about group properties related to the symmetric subshift and François Dahmani for sharing his thoughts concerning the soficness of the one-or-less subshift.

## References

- [1] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.
- [2] Nathalie Aubrun and Jarkko Kari. Tiling problems on baumslag-solitar groups. *Preprint*, 2013.
- [3] Nathalie Aubrun and Mathieu Sablik. Simulation of effective subshifts by two-dimensional subshifts of finite type. *Acta Applicandae Mathematicae*, 126:35–63, 2013.
- [4] Alexis Ballier and Maya Stein. The domino problem on groups of polynomial growth. *arXiv preprint arXiv:1311.4222*, 2013.
- [5] Robert Berger. *The Undecidability of the Domino Problem*. American Mathematical Society, 1966.
- [6] William W. Boone. The word problem. *Proceedings of the National Academy of Science USA*, 44:1061–1065, 1958.
- [7] Tullio Ceccherini-Silberstein and Michel Coornaert. *Cellular Automata and Groups*. Springer, 2009.
- [8] Aubrey da Cunha. Turing machines on cayley graphs. In LevD. Beklemishev and Ruy de Queiroz, editors, *Logic, Language, Information and Computation*, volume 6642 of *Lecture Notes in Computer Science*, pages 84–94. Springer Berlin Heidelberg, 2011.
- [9] François Dahmani. private communication.
- [10] François Dahmani and Asli Yaman. Symbolic dynamics and relatively hyperbolic groups. *Groups, Geometry, and Dynamics*, 2(2):165 – 184, 2008.

- [11] Bruno Durand, Andrei E. Romashchenko, and Alexander Shen. Effective closed subshifts in 1d can be implemented in 2d. In *Fields of Logic and Computation*, pages 208–226, 2010.
- [12] Anahí Gajardo and Jacques Mazoyer. One head machines from a symbolic approach. *Theoretical Computer Science*, 370(1-3):34 – 47, 2007.
- [13] Rostislav Grigorchuk. Degrees of growth of finitely generated groups, and the theory of invariant means. *Mathematics of the USSR-Izvestiya*, 1984.
- [14] Mikhail Gromov. Endomorphisms of symbolic algebraic varieties. *Journal of the European Mathematical Society*, 1(2):109–197, 1999.
- [15] Gustav Hedlund and Marston Morse. Symbolic dynamics. *American Journal of Mathematics*, 60(4):815–866, 1938.
- [16] Mike Hochman. On the dynamics and recursive properties of multidimensional symbolic systems. *Inventiones Mathematicae*, 176(1):131–167, 2009.
- [17] Mike Hochman and Tom Meyerovitch. A characterization of the entropies of multidimensional shifts of finite type. *Annals of Mathematics*, 171(3):2011–2038, 2010.
- [18] Douglas A. Lind and Brian Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1995.
- [19] Roger C. Lyndon and Paul E. Schupp. *Combinatorial group theory*. Springer-Verlag, 1977.
- [20] David E. Muller and Paul E. Schupp. Groups, the theory of ends, and context-free languages. *Journal of Computer and System Sciences*, 26(3):295 – 310, 1983.
- [21] Pyotr Novikov. On the algorithmic unsolvability of the word problem in group theory. *Proceedings of the Steklov Institute of Mathematics*, 44:143 pp. (Russian), 1955.
- [22] Alexander Yu. Ol’shanskii. An infinite group with subgroups of prime orders. *Mathematics of the USSR-Izvestiya*, 16(2):279, 1981.
- [23] Alexander Yu. Ol’shanskii and Mark V. Sapir. Non-amenable finitely presented torsion-by-cyclic groups. *Publications Mathématiques de l’IHÉS*, 96:43–169, 2002.
- [24] Michael O. Rabin. Recursive Unsolvability of Group Theoretic Problems. *Annals of Mathematics*, 67(1):pp. 172–194, 1958.
- [25] R. M. Robinson. Undecidable tiling problems in the hyperbolic plane. *Inventiones Mathematicae*, 44:159–264, 1978.

- [26] Raphael Robinson. Undecidability and nonperiodicity for tilings of the plane. *Inventiones Mathematicae*, 12:177–209, 1971.
- [27] John R. Stallings. On torsion-free groups with infinitely many ends. *Annals of Mathematics*, 88(2):pp. 312–334, 1968.
- [28] Benjamin Weiss. Sofic groups and dynamical systems. *Sankhyā, Ser. A*, 62(3):350–359, 2000.
- [29] Asli Yaman. private communication.

## A Appendix

In this section we give the proof omitted in the article.

**Proposition A.1.** *Let  $L$  be a language that can be decided by a multiple head  $G$ -machine. Then  $L$  can be decided by a  $G$ -machine*

*Proof.* We prove the result for a two heads  $G$ -machine, the generalization to an arbitrary number of heads follows easily. Suppose that a language  $L$  is decided by a two heads  $G$ -machine  $\mathcal{M}_2 = (Q, \Sigma, \sqcup, q_0, F, S, \delta)$ . We construct the one head  $G$ -machine  $\mathcal{M} = (Q', \Sigma', \sqcup', q'_0, F', S, \delta')$  as follows. We modify the alphabet in order to mark the original position – the identity  $1_G$  – of the two heads by a symbol  $\triangleright$ , and to store the positions  $h_1$  and  $h_2$  of the two heads in the group  $G$ , by marking the path from  $1_G$  to the first head (resp. second head) with symbols  $\times_i$ . The generator or generator inverse one should follow to find the next symbol  $\times_i$  is also attached to every  $\times_i$ . The new alphabet will be

$$\Sigma' = \Sigma^2 \times \underbrace{(S \cup S^{-1} \cup \{\sqcup\})^2}_{\text{direction to the next } \times_i} \times \underbrace{\{\times_1, \sqcup\} \times \{\times_2, \sqcup\}}_{\text{marked path to the heads}} \times \underbrace{\{\triangleright, \sqcup\}}_{\text{original position}} .$$

The new blank symbol will be  $\sqcup' = (\sqcup, \sqcup, \sqcup, \sqcup, \sqcup, \sqcup, \sqcup)$ . With this new alphabet, the first head (resp. second head) will be in position  $g$  in a configuration  $c$  if and only if  $c_g = (?, ?, \sqcup, ?, \times_1, ?, ?)$  (resp.  $c_g = (?, ?, ?, \sqcup, ?, \times_2, ?)$ ), in other terms position  $g$  is on the path from the identity  $1_G$  to the head, but this is the end of the path.

The head of the new machine will store the states of the two heads we want to simulate, and also uses its own states for searching for one of the head, moving back to the identity or updating the positions  $h_1$  and  $h_2$ . The new set of states will be

$$Q' = \{q_{\text{init}}^i, q_{\text{search}}^i, q_{\text{simul}}^i, q_{\text{id}}^i, q_{\text{update}}^i \mid i = 1, 2\} \times Q^2,$$

and we adopt the convention that the new initial state is  $q'_0 = (q_{\text{init}}, q_0, q_0)$ . The first step of the new machine will always consists in an initialization

$$\delta((q_{\text{init}}, q_0, q_0), (\sqcup, \sqcup, \sqcup, \sqcup, \sqcup, \sqcup, \sqcup)) = ((q_{\text{search}}^1, q_0, q_0), (\sqcup, \sqcup, \sqcup, \sqcup, \times_1, \times_2, \triangleright), 1_G),$$

which consist in marking the original position of the computation in the group with symbol  $\triangleright$  – this symbol will never be erased and appears only once – and stating that the first and second heads are initially both located at this original position. Then every transition of the two heads machine

$$\delta((q_1, q_2), (a_1, a_2)) = ((q'_1, q'_2), (b_1, b_2), (g_1, g_2))$$

will be decomposed into the following transitions – by convention, a symbol  $?$  represents any symbol that is not modified by the machine, and two symbols  $?$  in a transition rule do not necessarily represent the same symbol – in the one head machine



1. follow the path marked by  $\times_1$  until reaching position  $h_1$  where the first head is

$$\delta((q_{\text{search}}^1, q_1, q_2), (?, ?, h_1, ?, \times_1, ?, ?)) = ((q_{\text{search}}^1, q_1, q_2), (?, ?, h_1, ?, \times_1, ?, ?), h_1),$$

$$\delta((q_{\text{search}}^1, q_1, q_2), (?, ?, \sqcup, ?, \times_1, ?, ?)) = ((q_{\text{simul}}^1, q_1, q_2), (?, ?, \sqcup, ?, \times_1, ?, ?), 1_G),$$

2. simulate the first head and update  $h_1$

$$\delta((q_{\text{simul}}^1, q_1, q_2), (a_1, ?, \sqcup, ?, \times_1, ?, ?)) = ((q_{\text{update}}^1, q'_1, q_2), (b_1, ?, g_1, ?, \times_1, ?, ?), g_1),$$

$$\delta((q_{\text{update}}^1, q'_1, q_2), (?, ?, \sqcup, ?, \sqcup, ?, ?)) = ((q_{\text{id}}^1, q'_1, q_2), (?, ?, \sqcup, ?, \times_1, ?, ?), 1_G),$$

$$\delta((q_{\text{update}}^1, q'_1, q_2), (?, ?, g'_1, ?, \times_1, ?, ?)) = ((q_{\text{id}}^1, q'_1, q_2), (?, ?, \sqcup, ?, \times_1, ?, ?), 1_G),$$

3. go back to position  $\triangleright$

$$\delta((q_{\text{id}}^1, q'_1, q_2), (?, ?, g'_1, ?, \times_1, ?, \sqcup)) = ((q_{\text{id}}^1, q'_1, q_2), (?, ?, g'_1, ?, \times_1, ?, \sqcup), (g'_1)^{-1}),$$

$$\delta((q_{\text{id}}^1, q'_1, q_2), (?, ?, ?, ?, \times_1, ?, \triangleright)) = ((q_{\text{search}}^2, q'_1, q_2), (?, ?, ?, ?, \times_1, ?, \triangleright), 1_G),$$

4. follow the path marked by  $\times_2$  until reaching position  $h_2$  where the first head is

$$\delta((q_{\text{search}}^2, q'_1, q_2), (?, ?, ?, h_2, ?, \times_2, ?)) = ((q_{\text{search}}^2, q'_1, q_2), (?, ?, ?, h_2, ?, \times_2, ?), h_2),$$

$$\delta((q_{\text{search}}^2, q'_1, q_2), (?, ?, ?, \sqcup, ?, \times_2, ?)) = ((q_{\text{simul}}^2, q'_1, q_2), (?, ?, \sqcup, ?, \times_1, ?, ?), 1_G),$$

5. simulate the second head and update  $h_2$

$$\delta((q_{\text{simul}}^2, q'_1, q_2), (?, a_2, ?, \sqcup, ?, \times_1, ?)) = ((q_{\text{update}}^2, q'_1, q'_2), (?, a'_2, ?, g_2, ?, \times_2, ?), g_2),$$

$$\delta((q_{\text{update}}^2, q'_1, q'_2), (?, ?, ?, \sqcup, ?, \sqcup, ?)) = ((q_{\text{id}}^2, q'_1, q'_2), (?, ?, ?, \sqcup, ?, \times_2, ?), 1_G),$$

$$\delta((q_{\text{update}}^2, q'_1, q'_2), (?, ?, ?, g'_2, ?, \times_2, ?)) = ((q_{\text{id}}^2, q'_1, q'_2), (?, ?, ?, \sqcup, ?, \times_2, ?), 1_G),$$

6. go back to position  $\triangleright$

$$\delta((q_{\text{id}}^2, q'_1, q'_2), (?, ?, ?, g'_2, ?, \times_2, \sqcup)) = ((q_{\text{id}}^2, q'_1, q'_2), (?, ?, ?, g'_1, ?, \times_2, \sqcup), (g'_2)^{-1}),$$

$$\delta((q_{\text{id}}^2, q'_1, q'_2), (?, ?, ?, ?, ?, \times_2, \triangleright)) = ((q_{\text{search}}^1, q'_1, q'_2), (?, ?, ?, ?, ?, \times_2, \triangleright), 1_G),$$

By giving as set of final states

$$F' = \{q_{\text{init}}, q_{\text{search}}^i, q_{\text{simul}}^i, q_{\text{id}}^i, q_{\text{update}}^i \mid i = 1, 2\} \times F^2,$$

we ensure that the language decided by the  $G$ -machine  $\mathcal{M}$  is  $L$ .  $\square$