ENS Paris-Saclay – University of Turku

Research internship – ARPE

2021–2022

Symbolic dynamics and groups

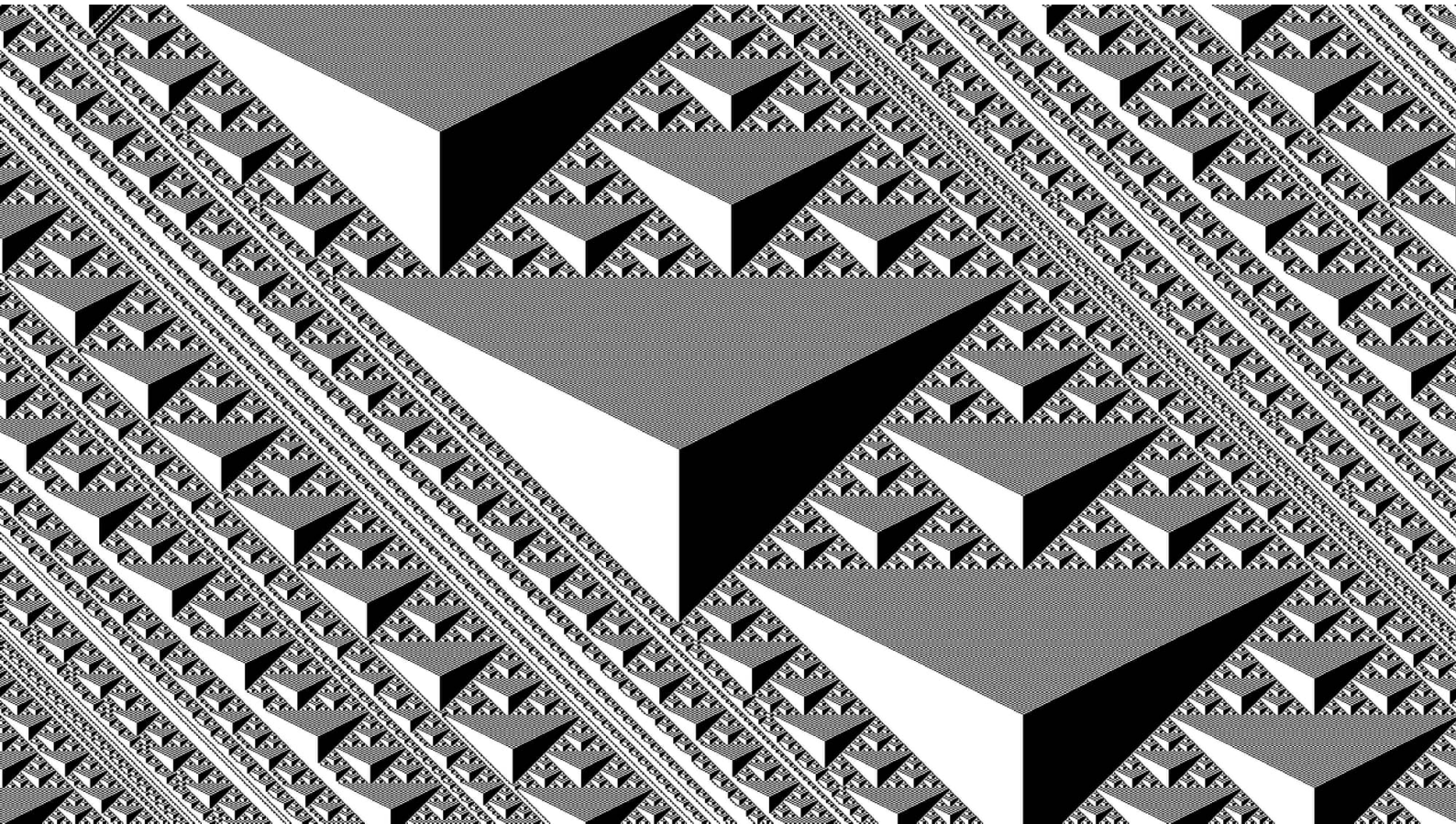
Author:
Antonin Callard

Advisor:
Ville Salo

école normale supérieure paris–saclay

UNIVERSITY OF TURKU

Space-time diagram, rule 4067213884 (biradius 5)
A New Kind of Science, Stephen Wolfram



Contents

Prerequisites, notations and conventions

Throughout this report, we assume the reader is somewhat familiar with some basic definitions of:

- computability theory: Turing machines, time and space complexity[1]...
- topology: openess, closeness, compactness...
- group theory: conjugacy, commutators, Cayley graphs, group extensions[2]...

Any other notion is (hopefully) explained where needed.

Notations

Here are a few notations that we use in this document:

$\#S$	Cardinal of a finite set S
$[\![i, j]\!]$	Interval $\{n \in \mathbb{N} : i \leq n \leq j\}$
$\mathrm{v}_k(w)$	Value of $w \in \{0, \dots, k-1\}^*$ in base k (big-endian)
$n_{(k)}$	Expression of $n \in \mathbb{N}$ in base k
Σ	A finite alphabet
Σ^n	Words of length n over Σ
Σ^*	Words over Σ
$\overline{w}$	Reverse (or "mirror") of $w \in \Sigma^*$
$w\|_J$	Restriction of w to $J \subseteq [\![0, \mathrm{len}(w) - 1]\!]$
$\mathcal{S}$	The SMART machine
ℓ	Length of the tape
$T_{\ell,\mathcal{M}}$	Action of $\mathcal{M}$ on tapes of length ℓ
$f_{\mathcal{M}}$	Action of $\mathcal{M}$ on full-shifts (conveyor belts)
$h_m(X)$	m-entropy of X
$[w] \in G$	Composition morphism $[\cdot] : G^* \mapsto G$
$A \Subset B$	$A \subseteq B$, A locally compact in B

To be perfectly clear, if $w = w_0 \dots w_{n-1}$, then $\overline{w} = w_{n-1} \dots w_0$; and if $J = \{j_0, \dots, j_k\} \subseteq [\![0, n-1]\!]$, with $j_0 < j_1 \cdots < j_k$, then $w|_J = w_{j_0} \dots w_{j_k}$.

And for $w \in G^*$ a word over with letters in the group G, then $[w] = w_0 \cdot w_1 \dots w_{\mathrm{len}(w)-1} \in G$.

[1] In Lemma 2.4, we denote NC^1 for "Nick's Class" of complexity of level 1, i.e. the class of languages $L \subseteq \Sigma^*$ such that L is decidable by Boolean circuits with a polynomial number of gates, with at most two inputs and depth $O(\log n)$ (see for example [AB09]). The reader need not be familiar with this class to follow our argument.

[2] Group extensions are a key ingredient in Chapter 4, but we invite the unfamiliar reader to skip the mathematical technicalities of the chapter and focus on the informal descriptions and statements.

General conventions

We also use the following conventions:

$X, Y, Z, \ldots$	Subshifts
$x, y, z, \ldots$	Configurations
i	Position in $\mathbb{Z}$
$u, v, w, \ldots$	Patterns/words
$\mathrm{Aut}(X)$	Automorphisms of X
G	Groups
$\mathcal{M}$	Turing machines

Group theory

Our groups always act from the left. If $g_1, \ldots, g_n$ are commuting elements of a group, we write $\prod_{i=1}^{n} g_i$ for their ordered product $g_n \cdots g_1$. In groups of bijections on a set (which almost all our groups are), we denote composition by $\circ$.

For G a group, $g \in G$ and $S \subseteq G$ a finite set of elements, we denote by $|g|_S$ the word norm of g in the finitely generated subgroup $\langle S \rangle$, i.e. the length of the shortest presentation of g in terms of elements of S.

For a, b elements of a group, we use the following conventions for conjugation and commutators:

$$a^b = b^{-1}ab \qquad \text{(Conjugacy)}$$
$$[a, b] = a^{-1}b^{-1}ab \qquad \text{(Commutator)}$$

If $\pi \in \mathrm{Sym}(A)$ is a permutation, we may regard it as a permutation of $A \times B$ by $\pi((a, b)) = (\pi(a), b)$. Slightly abusing the English language, we say a permutation of $A \times B$ is *B-ignorant*, or *ignores B*, if it comes from a permutation of A through this formula.

"Magic's all numbers and angles and edges and what the stars are doing, as if that really mattered. It's all power. It's all–" Granny paused, and dredged up her favourite word to describe all she despised in wizardry, "–jommetry."
— Terry Pratchett, Equal Rites

"So, ice-swimming on Monday?"
— Ville Salo

General Introduction

General context

Subshifts are sets of colorings (or *configurations*) defined by families of forbidden patterns. While they were originally introduced as a way to discretize continuous dynamical systems by cutting their phase spaces into different colors and studying all the iterations of points by a transformation, these mathematical objects started to be studied for themselves and the name *subshift* appeared.

Over the last two decades, the convergence of the mathematical and computer science communities has brought group theory as a tool to study the behavior of subshifts, and provide some generalizations. More precisely, two main directions of study exist:

1. Given a subshift X, its *automorphisms* (the set of homeomorphisms between X and itself, i.e. the continuous and bijective recoloring functions that act on X) define a group $\mathrm{Aut}(X)$ for composition. The structure of $\mathrm{Aut}(X)$ can be quite rich and complex, and the main direction of work studies which groups can or cannot embed as subgroups of automorphism groups.
2. Given a group G, one can define subshifts using G as a surface, by considering sets of colorings of G which do not contain some families of forbidden patterns. The geometry of the group G can then influence the behavior of the subshifts of surface G, in particular in terms of aperiodicity.

In this internship, I aim at exploring further the relationship between symbolic dynamics and groups. In particular, I independently focused on two problems: the existence of distortion in automorphism groups of subshifts, and the influence of the geometry of a subshift surface on aperiodic configurations.

Distortion in automorphism groups of subshifts

As explained above, a sizeable direction of study considers which groups embed (or cannot embed) in automorphism groups of one-dimensional subshifts, i.e. colorings of the infinite discrete line $\mathbb{Z}$ with symbols on some alphabet Σ (with some patterns being forbidden).

Distortion is a group-theoretic notion that appears in many classical groups (Baumslag-Solitar groups, the Heisenberg group...) whose embedding as subgroups of automorphism groups are still open problems. For distortion-related works in the context of group theory, see [Gro93; CF06; GK11; LM18; GL19; CC20; FH06; Pen20; Nav21]. In the context of automorphism groups of subshifts, distortion was proved to be an obstacle to embeddings in some cases, see [CFKP18, Corollary 3.10].

The existence of distortion in automorphism groups of subshifts has been an open problem since at least [KR90; CFKP18]. During this internship, we proved that distortion appears in some automorphism groups, and obtained the following results, which we detail in Chapters 1 to 3:

- We study the behavior of the SMART machine [COT17] on finite cyclic tapes.
- (A slightly altered of) the SMART machine defines a distorted automorphism on some full-shift.
- As a corollary, sofic subshifts admit distorted automorphisms if and only if their entropy is non-zero.

While our results are stated in terms of automorphisms on subshifts, our proof methods rather belong to the theories of dynamical Turing machines and reversible gates. For this reason, we split this work in several chapters. Chapter 1 defines dynamical Turing machines, and studies the recursive behavior of SMART machine from [COT17]. Chapter 2 uses results about permutations in the reversible gate model to prove what we call the "finitary distortion of the SMART machine". Finally, Chapter 3 contains all our considerations on subshifts, their automorphism groups, distortion inside them, and a few interesting corollaries.

Group confluence

Completely independently, we also studied the influence of the geometry of a group G on subshifts with surface G, motivated by previous work I did with Benjamin Hellouin de Menibus [CH22, Proposition 25] about aperiodicity in $\mathbb{Z}^d$ subshifts. We had proved that, if the so-called "surface entropy" of a sofic subshift X was infinite, then X must contain an aperiodic configuration. But this property was restricted to sofic subshifts, and the general case was still escaping us a few months after the publication of the original paper.

Ville Salo joined us in this study, and together we achieved the following, which we detail in Chapter 4:

- Any $\mathbb{Z}^d$ subshift of infinite surface entropy must contain a (strongly) aperiodic configuration.
- We define the notion of *group confluence*, which is implicitly used in the proof of the first item. We then prove the confluence of some classes of groups.
- It follows that subshifts with infinite surface entropy on such groups have aperiodic configurations.

Future prospects

Exhibiting a distortion element in an automorphism group naturally leads to more questions. Which rates of distortion can exist, cannot? Can the Baumslag-Solitar groups embed in automorphism groups of subshifts? What about the Heisenberg group? As our construction is quite convoluted, does there exist a more natural example? For more details, see Section 3.6 – "open questions".

In terms of group geometry and aperiodicity, our the definition of surface entropy with groups uses arbitrary shapes for patterns, while on $\mathbb{Z}^d$ it uses hypercubes: we would like to consider this gap before we post any draft online. Of course, one obvious question would be: does the confluence property hold for arbitrary torsion-free solvable groups? It very-well may be the case, but building groups by induction won't prove it. For more details, see Section 4.2.4 – "open questions".

Meta-information

This internship was very enlightening. It was my first time as an expat, and being abroad for such a long time – discovering new people and a new culture – was quite an experience!

Mathematically, when I asked Ville Salo to be my ARPE supervisor (following recommendations of my future PhD advisor, Pascal Vanier), we picked this topic halfway between symbolic dynamics and group theory: I was already a bit familiar with the former, since two of my previous internships were already about computability considerations on subshifts; however, I was completely ignorant of group theory above an undergraduate level, and I had some catching up to do[1].

In October, we studied the finite behavior[2] of the SMART machine and created the bijection that encodes finite cyclic configurations into ternary numbers (Chapter 2). From November to the end December, we focused on implementing our ideas in the reversible gate model, which was probably the most difficult part of this internship. The *ducking trick* was our first breakthrough, which lead to the results of Lemma 2.1 in Chapter 2. Back in France during the winter break, I met with Benjamin Hellouin to once again fail to generalize our conjecture about entropies and aperiodicities in $\mathbb{Z}^d$ subshifts. Talking about these attempts with Ville would later be useful.

Back in Finland, universities were closed from mid-January to mid-February and the Finnish winter was rather cold and lonely. I remotely attended lectures about cellular automata and did some exercise sessions for fun, but couldn't bring myself to do research. At the end of February, universities reopened, and we obtained our first example of a distortion element (in a very specific subshift). These results were included in the midterm report I was writing at that time[3], and implemented in Python.

[1] After a year of study, I realize that, of course, I still don't know anything about group theory; but hopefully, I'm a little less ignorant than before.

[2] We studied the behavior of finite cyclic tapes as we were convinced we could then move these results into a distortion element in some full-shift.

[3] Since then, we ditched these results, as we generalized our construction to any non-trivial full-shift.

During the last days of March, we came up with the *two-scale trick*, which lead in April to the key proof of Chapter 3 and to exhibiting a distortion element in some full-shift (Lemma 3.9). From April to the end of July, we compiled all of our findings in an article [CS22]: as the proofs were quite technical, we went back and forth several times. This wasn't my first time writing an article, but this one was probably the hardest. In parallel, I mentioned back in May to Ville Salo our (failed) attempts with Benjamin Hellouin to generalize [CH22, Proposition 25] about surface entropies; a few cups of coffee later, some emails bounced back and forth, and we worked until the end of June to prove the group geometry results of Chapter 4.

I was in Caen for a seminar (visiting Pascal Vanier, my future PhD advisor) during a few days in June, and then in Liège for the SDA2 meeting (the annual meeting of the French-speaking symbolic dynamics community). I then presented my results about distortion at a seminar in Turku. Back in Turku from mid-June to mid-July, the little time we had left was spent writing [CS22] (which took most of it), implementing Chapter 2 in Python[4], and of course doing some side-activities (saunas, biking, hiking…). After I left Turku, we completed our draft, published it online and gathered some comments and discussions.

Acknowledgements

Of course, I am deeply indebted to Ville Salo for his friendly, joyful and *SMART* supervision. Even after a whole year, and despite my eagerness to learn from his bottomless pits of knowledge, I still have to find a single topic in which he isn't more knowledgeable than me. As a result, I asked him many very stupid questions about subshifts, group theory and math in general, and he was unbelievably patient and kind to not point them as such while answering them all. He also made me feel at home in Finland, planning my arrival, making me try Finnish *salmiakki*[5], *runebergintorttu*[6], *mämmi*[7], delicious *lohikeitto*[8] and other surprising Finnish food. We also went skiing[9], and of course he brought me to my first Finnish saunas. The latter go, without saying, with ice-swimming[10]: I had never thought I would try such a terrifying hobby, and it definitely hadn't crossed my mind I could like it, but… here I am!

Of course, I want to thank the people I met in Finland: Anni, Serpi, Jarkko and Ilkka, who indulged with me in coffee discussions or welcomed me in their lectures when I felt a bit lonely during the Finnish winter; Liisa and the people of the Finnish lectures; the students of Delta, who welcomed the non-Finnish speaker I was in their monthly game nights. Without saying, my special thanks go to Juliette and Hervé, the two French co-interns who arrived here at the end of February: Juliette, for being the most "normalienne" person I ever met, supporting every crazy last-minute plan and idea I could throw at her and suggesting many others; and Hervé, for the discussions, the movies, and the video games we played late at night[11]. You have been amazing cooks, raclette-enjoyers, sushi-fanatics, series binge-watchers, co-players, co-hikers, co-bikers and friends, witnessing with me the estranging Finnish way of life. I hope we'll meet again in similar circumstances.

Finally, my gratitude goes to a few special relatives who were in touch despite the distance: my parents, who've been there for me the whole time; Durab, for the weekly *Don't Starve* and movie nights, our visits of Stockholm and Riga, and for calling again and again to keep me company; and of course Mwette, for the immense support, the friendship, the tears, the laughs, and *everything* she did for me this whole year. This piece of work wouldn't exist if it weren't for them.

Last but not least: I wish you a great and enjoyable reading, and I hope these pages will, between the lines, let you catch a glimpse of the memories I gained in Finland!

[4] The code is not ready for publication, at it contains some bugs and has no documentation. I plan to clean this at some point in the future (counterpoint: most of my plans stay hypothetical for undefined amounts of time). Anyway, despite a few bugs, we consider it a proof of concept.

[5] Finnish salty licorice, presumably the "weird food" strangers have to try. But I quite liked them!

[6] Finnish desert in February, has an amazing taste.

[7] Finnish desert around Easter, mainly made of rye: as Ville said, it really does taste like "black stuff".

[8] Finnish salmon soup, with quite an unhealthy but delicious amount of cream.

[9] Cross-country skiing, obviously. Ville said I was better than four-year olds at this, which is still a compliment, I guess?

[10] My guess is that Ville wanted to hear me scream my lungs out, and he got what he asked for…

[11] During which *Outer Wilds* scared me to death at least as much as it scared you!

1

The SMART machine on cyclic tapes

In this report, a Turing machine is a dynamical system where a single head moves over an infinite tape of arbitrary data (over a fixed finite alphabet), and all the action happens near the head (which may move around the tape, such movement depending on the content of the tape; or modify said content). The dynamics of Turing machines is an important branch of symbolic dynamics, initiated in [Kůr97], which explicitly defined the moving-head and moving-tape dynamics of Turing machines (although dynamical ideas appeared in the literature before, see [Hoo66; Rog75; Moo91]).

One of the most-studied behaviors of Turing machines is aperiodicity, meaning that the action of the Turing machine has no periodic points. An aperiodic Turing machine was exhibited in [BCN02] (inspired by techniques from [Hoo66]), and [KO08] found the first reversible aperiodic Turing machines (ones whose action is a homeomorphism). This culminated in the discovery of the SMART machine $\mathcal{S}$ [COT17], a machine with only four states and three tape-letters, which is reversible and aperiodic, and whose moving-tape dynamics is a minimal homeomorphism on the Cantor space (see also [Oll18]).

For the purpose of establishing distortion, the first important consideration is the "speed" of a Turing machine: a Turing machine with linear movement, meaning the existence of tape contents such that the head moves to infinity at a positive rate, could not possibly give rise to a distortion element. However, the SMART machine $\mathcal{S}$ can only move by an offset of at most $O(\log t)$ in t steps of computation. This makes $\mathcal{S}$ a perfect candidate for a distortion element, as conjectured in [GS17].

This chapter focuses on the dynamics of general considerations about the SMART machine. We underline the recursive aspects of its computations, and consider its action on finite cyclic tapes. As it contains no consideration about subshifts, automorphisms or distortion, this chapter is somehow independent of the rest of this report, even though Chapter 2 heavily relies on the encoding defined in Section 1.3.

1.1 Definitions

Let Q be a finite set called the *state set*, and Γ be a finite set called the *tape alphabet*. In the model[1] of [KO08], a *Turing machine* is a triple $\mathcal{M} = (\Gamma, Q, \Delta)$, where $\Delta \subseteq (Q \times \{+1, -1\} \times Q) \cup (Q \times \Gamma \times Q \times \Gamma)$ is the *transition table*. A transition $(q, \delta, q') \in Q \times \{+1, -1\} \times Q$ is called a *move transition*, and a transition $(q, a, q', b) \in Q \times \Gamma \times Q \times \Gamma$ is called a *matching transition*.

[1] This model is equivalent to the usual definition of Turing machines, but handles reversibility better.

In this report, we focus on the action of Turing machines on two objects: bi-infinite tapes, and finite cyclic tapes.

Bi-infinite tapes

On the alphabet $\Gamma \cup (Q \times \Gamma)$, elements of $Q \times \Gamma$ are called *heads*. Denote

$$X_{Q,\Gamma} = \{x \in (\Gamma \cup (Q \times \Gamma))^{\mathbb{Z}} \mid \forall i, j \in \mathbb{Z} : i \neq j \implies x_i \in \Gamma \vee x_j \in \Gamma\}$$

the set of bi-infinite tapes with at most one head somewhere. We can associate to $\mathcal{M}$ its so-called *moving-head model* [Kůr97], i.e. the binary relation $\to_{\mathcal{M}}$ on $X_{Q,\Gamma}$ defined by $x \to_{\mathcal{M}} x$ if $x \in \Gamma^{\mathbb{Z}}$, and if $x_i = (q, a_i)$ for $i \in \mathbb{Z}$, then $x \to_{\mathcal{M}} x'$ if there exists $t \in \Delta$ such that:

$$\text{If } t = (q, a_i, q', b) \in \Delta : \qquad x'_j = \begin{cases} (q', b) & \text{if } j = i \\ x_j & \text{otherwise} \end{cases}$$

$$\text{If } t = (q, \delta, q') \in \Delta : \qquad x'_j = \begin{cases} a_i & \text{if } j = i \\ (q', x_{i+\delta}) & \text{if } j = i + \delta \\ x_j & \text{otherwise} \end{cases}$$

The machine $\mathcal{M}$ is *deterministic* if $\to_{\mathcal{M}}$ defines a partial function, *complete deterministic* if it defines a total function (which is then continuous and, obviously, shift-commuting), and *complete reversible* (or *reversible* for short) if it defines a bijection (which is then a homeomorphism). When $\mathcal{M}$ is complete deterministic (which all our machines are), when using the relation $\to_{\mathcal{M}}$ as a function we write it as $T_{\mathcal{M}} : X_{Q,\Gamma} \mapsto X_{Q,\Gamma}$, which is an endomorphism of the subshift $X_{Q,\Gamma}$. Similarly, when $\mathcal{M}$ is reversible, it is an automorphism of $X_{Q,\Gamma}$.

Finite cyclic tapes

The set of *cyclic configurations of length* ℓ is the set $C_{\ell,Q,\Gamma} = \mathcal{L}_\ell(X_{Q,\Gamma})$ of finite configurations containing at most one head. Then $\mathcal{M}$ defines a binary relation $\to_{\mathcal{M}}$ on $C_{\ell,Q,\Gamma}$ by considering these finite tapes cyclic, i.e. we define $x \to_{\mathcal{M}} x$ if $x \in \Gamma^\ell$, and if $x_i = (q, a_i)$ for $i \in [\![0, \ell - 1]\!]$, then $x \to_{\mathcal{M}} x'$ if there exists $t \in \Delta$ such that:

$$\text{If } t = (q, a_i, q', b) \in \Delta : \qquad x'_j = \begin{cases} (q', b) & \text{if } j = i \\ x_j & \text{otherwise} \end{cases}$$

$$\text{If } t = (q, \delta, q') \in \Delta : \qquad x'_j = \begin{cases} a_i & \text{if } j = i \\ (q', \pi_\Gamma(x_{i+\delta \bmod l})) & \text{if } j = i + \delta \bmod \ell \\ x_j & \text{otherwise} \end{cases}$$

where $\pi_\Gamma : \Gamma \sqcup (Q \times \Gamma) \mapsto \Gamma$ is the natural projection. If $\mathcal{M}$ is complete deterministic, the function $\to_{\mathcal{M}}$ will be denoted by $T_{\ell,\mathcal{M}} : C_{\ell,Q,\Gamma} \mapsto C_{\ell,Q,\Gamma}$. Note that it is an endomorphism of the shift action of $\mathbb{Z}$ (or $\mathbb{Z}_\ell$ which translates the cyclic tape around).

These conditions (determinism, completeness and reversibility) are characterized by obvious combinatorial properties. In particular, $\mathcal{M} = (\Gamma, Q, \Delta)$ is complete deterministic if and only if exactly one transition applies at any time:

$$\forall (q, a) \in (Q \times \Gamma) : \#\{t \in \Delta \mid t = (q, a, \cdot, \cdot) \text{ or } t = (q, \cdot, \cdot)\} = 1.$$

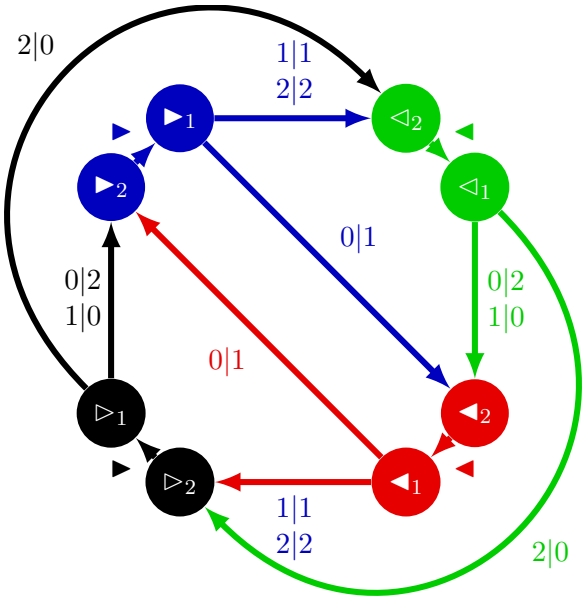
Defining the reverse of a transition by $(q, \delta, q')^{-1} = (q', -\delta, q)$ and $(q, a, q', b)^{-1} = (q', b, q, a)$, this reverse relation extends to transition tables with $\Delta^{-1} = \{t^{-1} \mid t \in \Delta\}$: the *reverse* of $\mathcal{M}$ is then defined by $\mathcal{M}^{-1} = (\Gamma, Q, \Delta^{-1})$, and $\mathcal{M}$ is reversible if both $\mathcal{M}$ and $\mathcal{M}^{-1}$ are complete deterministic.

Finally, for any machine $\mathcal{M} = (Q, \Gamma, \Delta)$, denote $m : \mathbb{N} \mapsto \mathbb{N}$ its *movement function*, i.e. $m(n)$ is the maximal number of cells the machine can visit in n steps.

1.2 The SMART machine

1.2.1 Definition and recursive moves

Let SMART be the Turing machine (Q, Γ, Δ), where $Q = \{\blacktriangleright_1, \blacktriangleleft_1, \triangleright_1, \triangleleft_1\} \cup \{\blacktriangleright_2, \blacktriangleleft_2, \triangleright_2, \triangleleft_2\}$, $\Gamma = \{0, 1, 2\}$ and Δ is the following transition table:



We refer to $\blacktriangleright_1\blacktriangleright_2, \blacktriangleleft_1, \blacktriangleleft_2$ (resp. $\triangleright_1, \triangleright_2, \triangleleft_1, \triangleleft_2$) as *filled* and *hollow triangles*.

Remark. *The SMART machine was introduced with a slightly different formalism in [COT17], and slightly revised in [Oll18] (states were renamed and permuted). The machine above adapts the latter in the model of [KO08] for Turing machines: in other words, we duplicate the states. We kindly advise readers already familiar with the SMART machine to read these definitions and propositions carefully.*

Namely, while our SMART machine is in a sense completely equivalent, in the formulas in Proposition 1.1 describing traversals of SMART over zeroes, the patterns corresponding to filled and hollow initial states are of the same length (unlike the corresponding ones in [COT17]). This will be helpful later, when we encode the position in the sweep into the corresponding area on the tape without any extra space.

Proposition 1.1

(Adapted from [COT17, Lemma 1]) Let $f(k) = 3^{k+1} - 2$.
For all $k \in \mathbb{N}$, $s_* \in \{0, 1, 2\}$ and $s_+ \in \{1, 2\}$, the following moves hold:

$$M_{\blacktriangleright}(k) : \begin{pmatrix} s_+ & 0^k & s_* \\ \blacktriangleright_2 & & \end{pmatrix} \vdash^{f(k)} \begin{pmatrix} s_+ & 0^k & s_* \\ & & \blacktriangleright_1 \end{pmatrix} \qquad M_{\blacktriangleleft}(k) : \begin{pmatrix} s_* & 0^k & s_+ \\ & & \blacktriangleleft_2 \end{pmatrix} \vdash^{f(k)} \begin{pmatrix} s_* & 0^k & s_+ \\ \blacktriangleleft_1 & & \end{pmatrix}$$

$$M_{\triangleright}(k) : \begin{pmatrix} s_* & 0^k & s_+ \\ \triangleright_2 & & \end{pmatrix} \vdash^{f(k)} \begin{pmatrix} s_* & 0^k & s_+ \\ & & \triangleright_1 \end{pmatrix} \qquad M_{\triangleleft}(k) : \begin{pmatrix} s_+ & 0^k & s_* \\ & & \triangleleft_2 \end{pmatrix} \vdash^{f(k)} \begin{pmatrix} s_+ & 0^k & s_* \\ \triangleleft_1 & & \end{pmatrix}$$

Additionally, the cell containing s_* is only visited at the last (resp. first) step of the sequences of transitions $M_{\blacktriangleright}$ and $M_{\blacktriangleleft}$ (resp. $M_{\triangleright}$ and $M_{\triangleleft}$). And the cell containing s_+ is never modified.

Proof. This proof adapts the proof of [COT17, Lemma 1], and highlights the recursive/nested aspects of these moves. In the case $k = 0$ one can check that indeed the formula describes a single transition. We

reason by induction, and assume $M_{\blacktriangleright}(k)$, $M_{\blacktriangleleft}(k)$, $M_{\triangleright}(k)$ and $M_{\triangleleft}(k)$ hold. We only prove $M_{\blacktriangleright}(k+1)$ and $M_{\triangleright}(k+1)$, by symmetry between $\blacktriangleright$ and $\blacktriangleleft$ (resp. $\triangleright$ and $\triangleleft$). Since $f(k+1) = 3f(k)+4$ we should find 3 recursions, and 4 extra steps. This is what happens:

$M_{\blacktriangleright}(k+1)$		$M_{\triangleright}(k+1)$	
	$\begin{pmatrix} s_+ & 0^k & 0 & s_* \\ \blacktriangleright_2 & & & \end{pmatrix}$		$\begin{pmatrix} s_* & 0 & 0^k & s_+ \\ \triangleright_2 & & & \end{pmatrix}$
	Apply $M_{\blacktriangleright}(k)$		Apply one step
$\vdash^{f(k)}$	$\begin{pmatrix} s_+ & 0^k & 0 & s_* \\ & & \blacktriangleright_1 & \end{pmatrix}$	$\vdash$	$\begin{pmatrix} s_* & 0 & 0^k & s_+ \\ & \triangleright_1 & & \end{pmatrix}$
	Apply one step		Apply one step
$\vdash$	$\begin{pmatrix} s_+ & 0^k & 1 & s_* \\ & & \blacktriangleleft_2 & \end{pmatrix}$	$\vdash$	$\begin{pmatrix} s_* & 2 & 0^k & s_+ \\ & \blacktriangleright_2 & & \end{pmatrix}$
	Apply $M_{\blacktriangleleft}(k)$		Apply $M_{\blacktriangleright}(k)$
$\vdash^{f(k)}$	$\begin{pmatrix} s_+ & 0^k & 1 & s_* \\ \blacktriangleleft_1 & & & \end{pmatrix}$	$\vdash^{f(k)}$	$\begin{pmatrix} s_* & 2 & 0^k & s_+ \\ & & & \blacktriangleright_1 \end{pmatrix}$
	Apply one step		Apply one step
$\vdash$	$\begin{pmatrix} s_+ & 0^k & 1 & s_* \\ \triangleright_2 & & & \end{pmatrix}$	$\vdash$	$\begin{pmatrix} s_* & 2 & 0^k & s_+ \\ & & & \triangleleft_2 \end{pmatrix}$
	Apply $M_{\triangleright}(k)$		Apply $M_{\triangleleft}(k)$
$\vdash^{f(k)}$	$\begin{pmatrix} s_+ & 0^k & 1 & s_* \\ & \triangleright_1 & & \end{pmatrix}$	$\vdash^{f(k)}$	$\begin{pmatrix} s_* & 2 & 0^k & s_+ \\ & \triangleleft_1 & & \end{pmatrix}$
	Apply one step		Apply one step
$\vdash$	$\begin{pmatrix} s_+ & 0^k & 0 & s_* \\ & & \blacktriangleright_2 & \end{pmatrix}$	$\vdash$	$\begin{pmatrix} s_* & 0 & 0^k & s_+ \\ & \triangleright_2 & & \end{pmatrix}$
	Apply one step		Apply $M_{\triangleright}(k)$
$\vdash$	$\begin{pmatrix} s_+ & 0^k & 0 & s_* \\ & & & \blacktriangleright_1 \end{pmatrix}$	$\vdash^{f(k)}$	$\begin{pmatrix} s_* & 0 & 0^k & s_+ \\ & & & \triangleright_1 \end{pmatrix}$

□

1.2.2 Action of SMART on cyclic tapes

This section studies the action of SMART on cyclic tapes of length $\ell \geq 2$. We call *initial configurations* the four following cyclic configurations:

$$C_{\blacktriangleright} = \begin{pmatrix} 0 & 0^{\ell-1} \\ \blacktriangleright_1 & \end{pmatrix} \qquad C_{\blacktriangleleft} = \begin{pmatrix} 0 & 0^{\ell-1} \\ \blacktriangleleft_1 & \end{pmatrix}$$
$$C_{\triangleright} = \begin{pmatrix} 0 & 0^{\ell-1} \\ \triangleright_1 & \end{pmatrix} \qquad C_{\triangleleft} = \begin{pmatrix} 0 & 0^{\ell-1} \\ \triangleleft_1 & \end{pmatrix}$$

Proposition 1.2

Let $\ell \geq 1$. The action of the $(2 \cdot 3^\ell)$-th power of SMART on $C_{\blacktriangleright}$ and $C_{\triangleright}$ (resp. $C_{\blacktriangleleft}$ and $C_{\triangleleft}$) is a right-shift (resp. left-shift). Furthermore, the intermediate configurations are all distinct.

Proof. By symmetries between $\blacktriangleright$ and $\blacktriangleleft$ (resp. $\triangleright$ and $\triangleleft$), we prove the result for $C_{\blacktriangleright}$ and $C_{\triangleright}$.

$$\begin{array}{ll}
& \begin{pmatrix} 0 & 0 & 0^{\ell-2} \\ & \blacktriangleright_1 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 1 & 0 & 0^{\ell-2} \\ & \blacktriangleleft_2 & \end{pmatrix} \\
& \text{Apply } M_{\blacktriangleleft}(\ell-1) \\
\vdash^{f(\ell-1)} & \begin{pmatrix} 1 & 0 & 0^{\ell-2} \\ & \blacktriangleleft_1 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 1 & 0 & 0^{\ell-2} \\ & \triangleright_2 & \end{pmatrix} \\
& \text{Apply } M_{\triangleright}(\ell-1) \\
\vdash^{f(\ell-1)} & \begin{pmatrix} 1 & 0 & 0^{\ell-2} \\ & \triangleright_1 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 0 & 0 & 0^{\ell-2} \\ & \blacktriangleright_2 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 0 & 0 & 0^{\ell-2} \\ & \blacktriangleright_1 & \end{pmatrix}
\end{array}
\qquad
\begin{array}{ll}
& \begin{pmatrix} 0 & 0 & 0^{\ell-2} \\ & \triangleright_1 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 2 & 0 & 0^{\ell-2} \\ & \blacktriangleright_2 & \end{pmatrix} \\
& \text{Apply } M_{\blacktriangleright}(\ell-1) \\
\vdash^{f(\ell-1)} & \begin{pmatrix} 2 & 0 & 0^{\ell-2} \\ & \blacktriangleright_1 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 2 & 0 & 0^{\ell-2} \\ & \triangleleft_2 & \end{pmatrix} \\
& \text{Apply } M_{\triangleleft}(\ell-1) \\
\vdash^{f(\ell-1)} & \begin{pmatrix} 2 & 0 & 0^{\ell-2} \\ & \triangleleft_1 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 0 & 0 & 0^{\ell-2} \\ & \triangleright_2 & \end{pmatrix} \\
& \text{Apply one step} \\
\vdash & \begin{pmatrix} 0 & 0 & 0^{\ell-2} \\ & \triangleright_1 & \end{pmatrix}
\end{array}$$

We used moves $M_{\blacktriangleright}(\ell-1), M_{\blacktriangleleft}(\ell-1), M_{\triangleright}(\ell-1)$ and $M_{\triangleleft}(\ell-1)$ in patterns that overlap themselves on their first and last letters in the cyclic tape. This is valid, because the cell containing s_* is only visited at the last (resp. first) step of $M_{\blacktriangleright}$ and $M_{\blacktriangleleft}$ (resp. $M_{\triangleright}$ and $M_{\triangleleft}$).

For the last claim, it is enough to show that the last (shifted) configuration does not appear before the last step. This is clear from looking at the first columns, which have positive values on all but the first step and the two last steps. □

Lemma 1.3

For $\ell \geq 1$, the action of SMART on cyclic tapes of length ℓ is composed of four disjoint cycles of length $2\ell \cdot 3^\ell$, which are the orbits of the four initial configurations. Additionally, the action of the $(2 \cdot 3^\ell)$-th power of SMART on a cyclic tapes consists in a right-shift (resp. left-shift) on the orbits of $C_{\blacktriangleright}$ and $C_{\triangleright}$ (resp. $C_{\blacktriangleleft}$ and $C_{\triangleleft}$).

Proof. By Proposition 1.2, the orbits of $C_{\blacktriangleright}$, $C_{\triangleright}$, $C_{\blacktriangleleft}$ and $C_{\triangleleft}$ are each of length $2\ell \cdot 3^\ell$ (number of shifts $\times$ number of steps for each shift). As there are $8\ell \cdot 3^\ell$ different cyclic configurations containing a head (eight different states with ℓ possible positions, and a ternary tape of length ℓ), we now only need to prove that the orbits of $C_{\blacktriangleright}$, $C_{\triangleright}$, $C_{\blacktriangleleft}$ and $C_{\triangleleft}$ are disjoint.

For this, observe again that in the proof of Proposition 1.2, the positive ternary letter 1 or 2 are never erased in the searches $M_{\blacktriangleright}(\ell-1)$, $M_{\blacktriangleleft}(\ell-1)$, $M_{\triangleright}(\ell-1)$ and $M_{\triangleright}(\ell-1)$. Considering the orbit of C_q for some fixed $q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\}$, the word 0^ℓ appears only at the first or last step of this proof (or their shifts): in particular, they are all in states either q_1 or q_2 (with this fixed q). This implies that these orbits cannot intersect. □

1.2.3 Analysis of SMART configurations

We now explain how, given a cyclic SMART configuration of length ℓ, we determine which orbit it belongs in and its position in this orbit, i.e. the number of steps required to obtain it from its corresponding initial configuration $C_{\blacktriangleright}, C_{\blacktriangleleft}, C_{\triangleright}$ or $C_{\triangleleft}$.

We say that a cyclic configuration is performing the j-th step of computation of $M_{\blacktriangleright}(k)$ (resp. $M_{\blacktriangleleft}(k)$, $M_{\triangleright}(k)$, $M_{\triangleleft}(k)$), for $0 \leq j \leq f(k)$, if it contains the j-th pattern of the sequence of transitions $M_{\blacktriangleright}(k)$ (resp. ...) of Proposition 1.1. At this point, it may not be clear that this is unique, but this will follow from our argument.

If a configuration is performing some step of computation from one of the moves $M_{\blacktriangleright}(k)$, $M_{\blacktriangleleft}(k)$, $M_{\triangleright}(k)$ or $M_{\triangleleft}(k)$, we refer to this move as its *computation of level* k.

Initialization

We call the following patterns *special patterns of level* $k \geq 1$:

$$s(\blacktriangleright_2, k) = \begin{pmatrix} s_+ & 0^{k-1} & 0 \\ & & \blacktriangleright_2 \end{pmatrix} \qquad s(\blacktriangleright_1, k) = \begin{pmatrix} s_+ & 0^{k} & s_* \\ & & \blacktriangleright_1 \end{pmatrix}$$

$$s(\blacktriangleleft_2, k) = \begin{pmatrix} 0 & 0^{k-1} & s_+ \\ \blacktriangleleft_2 & & \end{pmatrix} \qquad s(\blacktriangleleft_1, k) = \begin{pmatrix} s_* & 0^{k} & s_+ \\ \blacktriangleleft_1 & & \end{pmatrix}$$

$$s(\triangleright_2, k) = \begin{pmatrix} s_* & 0^{k} & s_+ \\ \triangleright_2 & & \end{pmatrix} \qquad s(\triangleright_1, k) = \begin{pmatrix} 0 & 0^{k-1} & s_+ \\ \triangleright_1 & & \end{pmatrix}$$

$$s(\triangleleft_2, k) = \begin{pmatrix} s_+ & 0^{k} & s_* \\ & & \triangleleft_2 \end{pmatrix} \qquad s(\triangleleft_1, k) = \begin{pmatrix} s_+ & 0^{k-1} & 0 \\ & & \triangleleft_1 \end{pmatrix}$$

By the proof of Proposition 1.1, we see that if a cyclic configuration contains a special pattern $s(\blacktriangleright_2, k)$, $s(\blacktriangleright_1, k)$, $s(\blacktriangleleft_2, k)$ or $s(\blacktriangleleft_1, k)$ (resp. $s(\triangleright_2, k)$, $s(\triangleright_1, k)$, $s(\triangleleft_2, k)$ or $s(\triangleleft_1, k)$), then it performs the last two steps of $M_{\blacktriangleright}(k)$ or $M_{\blacktriangleleft}(k)$ respectively (resp. the first two steps of $M_{\triangleright}(k)$ or $M_{\triangleleft}(k)$).

Claim 1.4

Given a cyclic configuration c of length ℓ containing a head, exactly one of the following holds:

- c is on the full-zero word 0^ℓ.
- c performs some step of computation of level 0 from either $M_{\blacktriangleright}(0)$, $M_{\blacktriangleleft}(0)$, $M_{\triangleright}(0)$ or $M_{\triangleleft}(0)$.
- there exists some unique $1 \leq k \leq \ell - 1$ such that c either performs the last two steps of $M_{\blacktriangleright}(k)$ or $M_{\blacktriangleleft}(k)$, or the first two steps of $M_{\triangleright}(k)$ or $M_{\triangleleft}(k)$.

Proof. The patterns of $M_{\blacktriangleright}(0)$, $M_{\blacktriangleleft}(0)$, $M_{\triangleright}(0)$ and $M_{\triangleleft}(0)$ (eight in total), along with the special patterns of every level, disjointly cover all the non-zero configurations with a head. □

Inductive analysis: $k \mapsto k + 1 \leq \ell - 1$

Assume a cyclic configuration of length ℓ is performing some computation of level k, for $k < \ell - 1$. Then it performs some computation of level $k + 1$, and Figure 1.1 details a case-analysis to figure out which computation it performs out of $M_{\blacktriangleright}(k + 1)$, $M_{\blacktriangleleft}(k + 1)$, $M_{\triangleright}(k + 1)$ or $M_{\triangleleft}(k + 1)$. (This analysis also implies that the computation step performed at a particular level is unique, if one exists.)

Conclusion:

This proves that every cyclic configuration of length ℓ is either on the full zero word, or performs the j-th step of computation of a move of level $\ell - 1$ for some $0 \leq j \leq f(\ell - 1)$. With the proof of Proposition 1.2, we then conclude whether the input configuration belongs in the orbit of $C_{\blacktriangleright}$, $C_{\blacktriangleleft}$, $C_{\triangleright}$ or $C_{\triangleleft}$, and its position in this orbit.

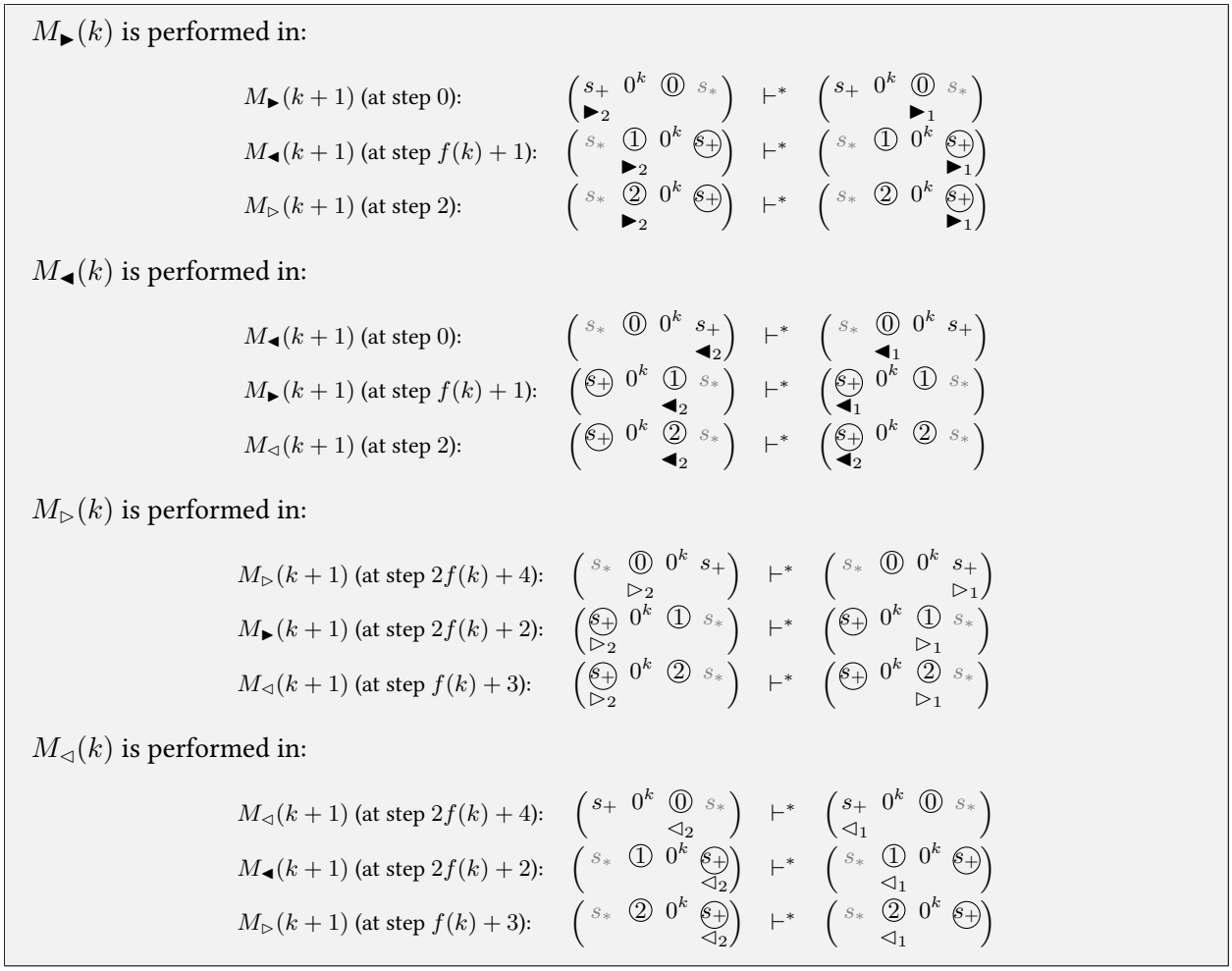


Figure 1.1: Bottom-up analysis of SMART configurations: $k \mapsto k+1$

We write configurations performing sub-computations (of level k) of computations of level $k+1$. We darken the part of the configuration performing the sub-computation of level k. Circled ternary bits are not modified by the sub-computations of level k, and can be used to perform a case-analysis.

1.3 Encoding cyclic tapes into their orbit positions in $C_{\ell,Q,\Gamma}$

Remark. *In this section, we work with the SMART machine introduced above; this will define encodings for the decorated machine (introduced below) by simply carrying the decorations in the state, unmodified.*

Denote $\mathcal{S}$ the SMART machine introduced above. We define $\Phi : C_{\ell,Q,\Gamma} \mapsto C_{\ell,Q,\Gamma}$ the following bijection:

$$\Phi\left(\mathcal{S}^n(C_q)\right) = \begin{cases} (q_1, w_1) \cdot w_2 \dots w_\ell, & \text{if } n < 3^\ell, \text{ for } w = n_{(3)} \\ & \text{and } q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\} \\ (q_2, w_1) \cdot w_2 \dots w_l, & \text{if } 3^\ell \leq n < 2 \cdot 3^l, \text{ for } w = [n - 3^\ell]_{(3)} \\ & \text{and } q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\} \\ \sigma^{-j}\left(\Phi\left(\mathcal{S}^{n'}(C_q)\right)\right) & \text{where } q \in \{\blacktriangleright, \triangleright\}, \quad 2j \cdot 3^\ell + n' = n \\ \sigma^{j}\left(\Phi\left(\mathcal{S}^{n'}(C_q)\right)\right) & \text{where } q \in \{\blacktriangleleft, \triangleleft\}, \quad 2j \cdot 3^\ell + n' = n \end{cases}$$

In other words, given as input a configuration $\mathcal{S}^n(C_q)$ of length ℓ, for some $q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\}$, Φ encodes the tuple (q, n) in base $4 \cdot (2\ell \cdot 3^\ell)$: the head has state ranging in q_1 or q_2 for $q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\}$, the position of the head ranges from 0 to $\ell - 1$, and the ternary counter of length ℓ ranges from 0 to $3^\ell - 1$. Note that this bijection is shift-commuting, because on each tape C_q, applying SMART $2 \cdot 3^\ell$ times performs a shift map (in the same direction as we do in the above formula).

Inductive encoding

In this section, we use the analysis performed in Section 1.2.3 to provide a linear-time algorithm that breaks this complicated bijection into smaller and easier steps, which we then "implement" in Section 2.2 in some finitely generated group of permutations.

Let c be a cyclic configuration of length ℓ of SMART, and $k \leq \ell - 1$. If c is not special of level $> k$ or on the full zero word, there exists some $q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\}$ and some unique word $w(k)$ of length $k + 2$ such that $w(k) \sqsubseteq c$ and $w(k)$ computes the j-th step of $M_q(k)$, for some $0 \leq j \leq f(k)$. For such k, we define the encoding of level k of c by replacing the occurrence of $w(k)$ in c by the word $w'(k)$ defined by:

If $q = \blacktriangleright$: $w'(k) = c_0 \dots c_k \cdot (q', w(k)_{k+1})$, where $c \in \{0, 1, 2\}^{k+1}$ is a ternary counter with $\mathrm{v}_3(c) = j + 1$, and $q' = \blacktriangleright_1$ if $w(k)_0 = 1$ (resp. $q' = \blacktriangleright_2$ if $w(k)_0 = 2$).

If $q = \blacktriangleleft$: $w'(k) = (q', w(k)_0) \cdot c_0 \dots c_k$, where $c \in \{0, 1, 2\}^{k+1}$ is a ternary counter with $\mathrm{v}_3(c) = j + 1$, and $q' = \blacktriangleleft_1$ if $w(k)_{k+1} = 1$ (resp. $q' = \blacktriangleleft_2$ if $w(k)_{k+1} = 2$).

If $q = \triangleright$: $w'(k) = (q', w(k)_0) \cdot c_0 \dots c_k$, where $c \in \{0, 1, 2\}^{k+1}$ is a ternary counter with $\mathrm{v}_3(c) = j + 1$, and $q' = \triangleright_1$ if $w(k)_{k+1} = 1$ (resp. $q' = \triangleright_2$ if $w(k)_{k+1} = 2$).

If $q = \triangleleft$: $w'(k) = c_0 \dots c_k \cdot (q', w(n)_{k+1})$, where $c \in \{0, 1, 2\}^{k+1}$ is a ternary counter with $\mathrm{v}_3(c) = j + 1$, and $q' = \triangleleft_1$ if $w(k)_0 = 1$ (resp. $q' = \triangleleft_2$ if $w(k)_0 = 2$).

Words with an all-zero counter value, i.e. $(0^{k+1} \cdot (\blacktriangleright_1, w_{k+1}))$, $(0^{k+1} \cdot (\blacktriangleright_2, w_{k+1}))$, $((\blacktriangleleft_1, w_0) \cdot 0^{k+1})$, $((\triangleleft_1, w_0) \cdot 0^{k+1})$ are not used in this encoding. Conveniently, they are exactly the special patterns of levels strictly greater than k.

Once we have specified the encodings at each level, we can define a sequence of transformations implicitly, by simply saying that we decode the level-k encoding and recode it on level $k + 1$; composing these for all values of k, we get the encoding of level ℓ, which one may easily check is simply Φ. In Figures 1.2, 1.3, 1.4 and 1.5, we perform the somewhat tedious exercise of describing these transformations in detail, as we need to know the form of these transformations in order to implement them with permutations later. Not surprisingly, it turns out that they require only ternary addition and basic rewriting of symbols.

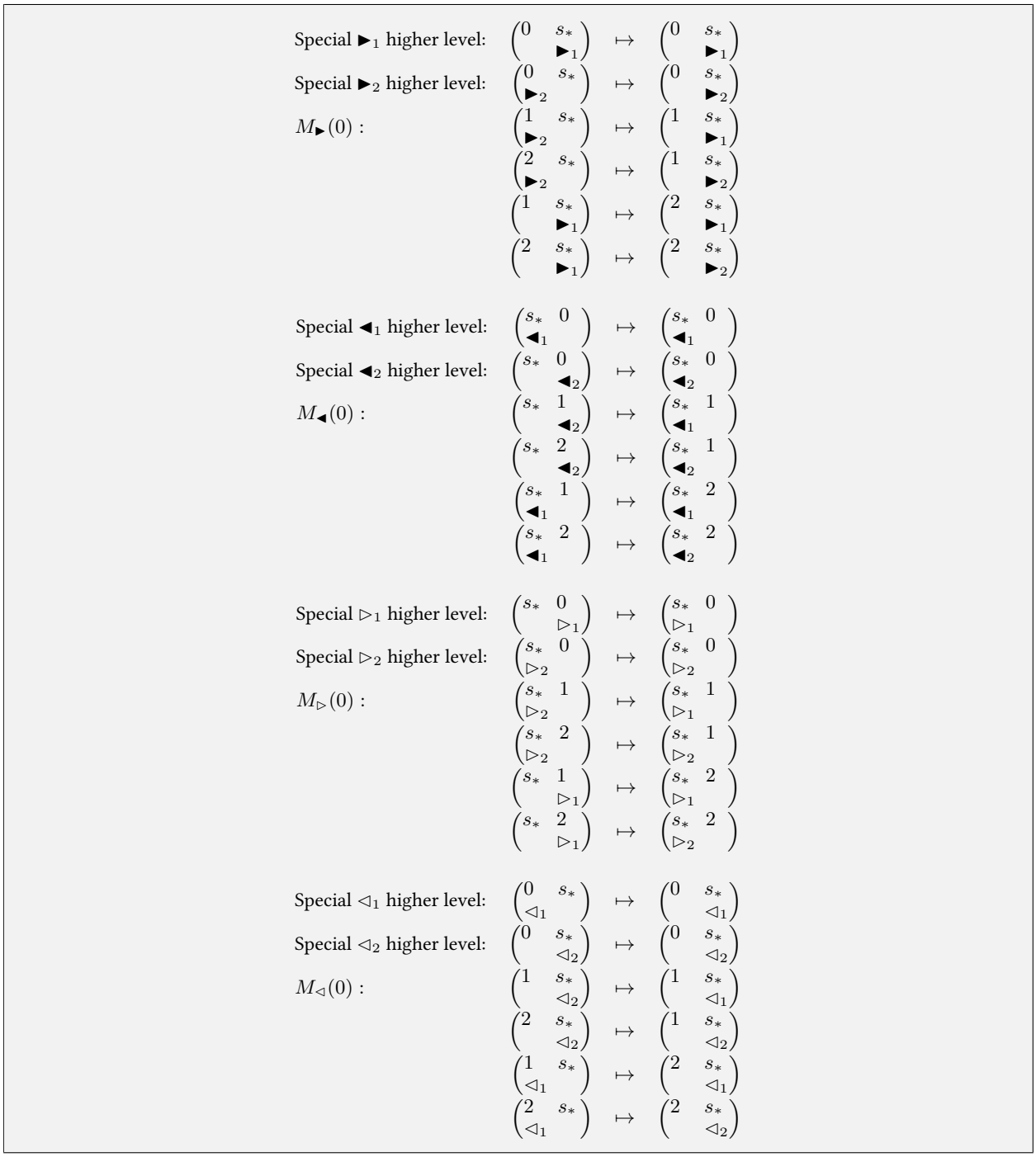


Figure 1.2: Encoding SMART configurations: $\Phi_{\text{init}} : c \mapsto c_0$

From a cyclic SMART configuration, compute its encoding of level 0. Rewriting words of length 2.

From $M_{\blacktriangleright}(k)$ to $(k+1)$-level encoding:

$$
\begin{array}{rcl}
M_{\blacktriangleright}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & c & 0 & * \\ & & \blacktriangleright_1 & \end{pmatrix} & \to \begin{pmatrix} * & 0 & c & * \\ & & & \blacktriangleright_1 \end{pmatrix} \\
& \begin{pmatrix} * & c & 0 & * \\ & & \blacktriangleright_2 & \end{pmatrix} & \to \begin{pmatrix} * & 0 & c & * \\ & & & \blacktriangleright_2 \end{pmatrix} \\
M_{\blacktriangleleft}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_1 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+1]_{(3)} & * \\ \blacktriangleleft_1 & & & \end{pmatrix} \\
& \begin{pmatrix} * & c & 2 & * \\ & & \blacktriangleright_1 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+1]_{(3)} & * \\ \blacktriangleleft_2 & & & \end{pmatrix} \\
M_{\triangleright}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_2 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2]_{(3)} & * \\ \triangleright_1 & & & \end{pmatrix} \\
& \begin{pmatrix} * & c & 2 & * \\ & & \blacktriangleright_2 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2]_{(3)} & * \\ \triangleright_2 & & & \end{pmatrix}
\end{array}
$$

From $M_{\blacktriangleleft}(k)$ to $(k+1)$-level encoding:

$$
\begin{array}{rcl}
M_{\blacktriangleleft}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & 0 & c & * \\ & \blacktriangleleft_1 & & \end{pmatrix} & \to \begin{pmatrix} * & 0 & c & * \\ \blacktriangleleft_1 & & & \end{pmatrix} \\
& \begin{pmatrix} * & 0 & c & * \\ & \blacktriangleleft_2 & & \end{pmatrix} & \to \begin{pmatrix} * & 0 & c & * \\ \blacktriangleleft_2 & & & \end{pmatrix} \\
M_{\blacktriangleright}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & 1 & c & * \\ & \blacktriangleleft_1 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+1]_{(3)} & * \\ & & & \blacktriangleright_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & c & * \\ & \blacktriangleleft_1 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+1]_{(3)} & * \\ & & & \blacktriangleright_2 \end{pmatrix} \\
M_{\triangleleft}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & 1 & c & * \\ & \blacktriangleleft_2 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2]_{(3)} & * \\ & & & \triangleleft_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & c & * \\ & \blacktriangleleft_2 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2]_{(3)} & * \\ & & & \triangleleft_2 \end{pmatrix}
\end{array}
$$

From $M_{\triangleright}(k)$ to $(k+1)$-level encoding:

$$
\begin{array}{rcl}
M_{\triangleright}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & 0 & c & * \\ & \triangleright_1 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+4]_{(3)} & \\ \triangleright_1 & & & \end{pmatrix} \\
& \begin{pmatrix} * & 0 & c & * \\ & \triangleright_2 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+4]_{(3)} & \\ \triangleright_2 & & & \end{pmatrix} \\
M_{\blacktriangleright}(k+1)(\text{if } 0\ \ c\neq 0): & \begin{pmatrix} * & 1 & c & * \\ & \triangleright_1 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+2]_{(3)} & \\ & & & \blacktriangleright_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & c & * \\ & \triangleright_1 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+2]_{(3)} & \\ & & & \blacktriangleright_2 \end{pmatrix} \\
M_{\triangleleft}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & 1 & c & * \\ & \triangleright_2 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+3]_{(3)} & * \\ & & & \triangleleft_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & c & * \\ & \triangleright_2 & & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+3]_{(3)} & * \\ & & & \triangleleft_2 \end{pmatrix}
\end{array}
$$

From $M_{\triangleleft}(k)$ to $(k+1)$-level encoding:

$$
\begin{array}{rcl}
M_{\triangleleft}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & c & 0 & * \\ & & \triangleleft_1 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+4]_{(3)} & \\ & & & \triangleleft_1 \end{pmatrix} \\
& \begin{pmatrix} * & c & 0 & * \\ & & \triangleleft_2 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+4]_{(3)} & \\ & & & \triangleleft_2 \end{pmatrix} \\
M_{\blacktriangleleft}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & c & 1 & * \\ & & \triangleleft_1 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+2]_{(3)} & \\ \blacktriangleleft_1 & & & \end{pmatrix} \\
& \begin{pmatrix} * & c & 2 & * \\ & & \triangleleft_1 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+2f(k)+2]_{(3)} & \\ \blacktriangleleft_2 & & & \end{pmatrix} \\
M_{\triangleright}(k+1)(\text{if } \mathrm{v}_3(c)\neq 0): & \begin{pmatrix} * & c & 1 & * \\ & & \triangleleft_2 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+3]_{(3)} & * \\ \triangleright_1 & & & \end{pmatrix} \\
& \begin{pmatrix} * & c & 2 & * \\ & & \triangleleft_2 & \end{pmatrix} & \to \begin{pmatrix} * & [\mathrm{v}_3(c)+f(k)+3]_{(3)} & * \\ \triangleright_2 & & & \end{pmatrix}
\end{array}
$$

Figure 1.3: Encoding SMART configurations: $\Phi_{k\mapsto k+1}$ (Part 1: $k \mapsto k+1$)

From an encoding of level k of a configuration, compute its $(k+1)$-encoding, following Figure 1.1. Cells with $*$ are unmodified, their point is to describe the head movement. We are rewriting words of length $k+4$ (including the two gray symbols). Note that at $k+1=\ell-2$ (resp. $k+1=\ell-1$), the $*$-cells overlap or each other (resp. the counter), because we reach the length of the cyclic tape.

Encoding special $M_{\blacktriangleright}(k+1)$:

$$\begin{array}{lll}
\text{Special } \blacktriangleright_1: & \begin{pmatrix} * & 1 & 0^{k+1} & * \\ & & & \blacktriangleright_1 \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)+1]_{(3)} & * \\ & & \blacktriangleright_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & 0^{k+1} & * \\ & & & \blacktriangleright_1 \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)+1]_{(3)} & * \\ & & \blacktriangleright_2 \end{pmatrix} \\
\text{Special } \blacktriangleright_2: & \begin{pmatrix} * & 1 & 0^{k+1} & * \\ & & & \blacktriangleright_2 \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)]_{(3)} & * \\ & & \blacktriangleright_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & 0^{k+1} & * \\ & & & \blacktriangleright_2 \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)]_{(3)} & * \\ & & \blacktriangleright_2 \end{pmatrix} \\
\text{Special } \blacktriangleright_1 \text{ higher level}: & \begin{pmatrix} * & 0 & 0^{k+1} & * \\ & & & \blacktriangleright_1 \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ & & \blacktriangleright_1 \end{pmatrix} \\
\text{Special } \blacktriangleright_2 \text{ higher level}: & \begin{pmatrix} * & 0 & 0^{k+1} & * \\ & & & \blacktriangleright_2 \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ & & \blacktriangleright_2 \end{pmatrix}
\end{array}$$

Encoding special $M_{\blacktriangleleft}(k+1)$:

$$\begin{array}{lll}
\text{Special } \blacktriangleleft_1: & \begin{pmatrix} * & 0^{k+1} & 1 & * \\ \blacktriangleleft_1 & & & \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)+1]_{(3)} & * \\ \blacktriangleleft_1 & & \end{pmatrix} \\
& \begin{pmatrix} * & 0^{k+1} & 2 & * \\ \blacktriangleleft_1 & & & \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)+1]_{(3)} & * \\ \blacktriangleleft_2 & & \end{pmatrix} \\
\text{Special } \blacktriangleleft_2: & \begin{pmatrix} * & 0^{k+1} & 1 & * \\ \blacktriangleleft_2 & & & \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)]_{(3)} & * \\ \blacktriangleleft_1 & & \end{pmatrix} \\
& \begin{pmatrix} * & 0^{k+1} & 2 & * \\ \blacktriangleleft_2 & & & \end{pmatrix} \to \begin{pmatrix} * & [f(k+1)]_{(3)} & * \\ \blacktriangleleft_2 & & \end{pmatrix} \\
\text{Special } \blacktriangleleft_1 \text{ higher level}: & \begin{pmatrix} * & 0^{k+1} & 0 & * \\ \blacktriangleleft_1 & & & \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ \blacktriangleleft_1 & & \end{pmatrix} \\
\text{Special } \blacktriangleleft_2 \text{ higher level}: & \begin{pmatrix} * & 0^{k+1} & 0 & * \\ \blacktriangleleft_2 & & & \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ \blacktriangleleft_2 & & \end{pmatrix}
\end{array}$$

Encoding special $M_{\triangleright}(k+1)$:

$$\begin{array}{lll}
\text{Special } \triangleright_1: & \begin{pmatrix} * & 0^{k+1} & 1 & * \\ \triangleright_1 & & & \end{pmatrix} \to \begin{pmatrix} * & [2]_{(3)} & * \\ \triangleright_1 & & \end{pmatrix} \\
& \begin{pmatrix} * & 0^{k+1} & 2 & * \\ \triangleright_1 & & & \end{pmatrix} \to \begin{pmatrix} * & [2]_{(3)} & * \\ \triangleright_2 & & \end{pmatrix} \\
\text{Special } \triangleright_2: & \begin{pmatrix} * & 0^{k+1} & 1 & * \\ \triangleright_2 & & & \end{pmatrix} \to \begin{pmatrix} * & [1]_{(3)} & * \\ \triangleright_1 & & \end{pmatrix} \\
& \begin{pmatrix} * & 0^{k+1} & 2 & * \\ \triangleright_2 & & & \end{pmatrix} \to \begin{pmatrix} * & [1]_{(3)} & * \\ \triangleright_2 & & \end{pmatrix} \\
\text{Special } \triangleright_1 \text{ higher level}: & \begin{pmatrix} * & 0^{k+1} & 0 & * \\ \triangleright_1 & & & \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ \triangleright_1 & & \end{pmatrix} \\
\text{Special } \triangleright_1 \text{ higher level}: & \begin{pmatrix} * & 0^{k+1} & 0 & * \\ \triangleright_2 & & & \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ \triangleright_2 & & \end{pmatrix}
\end{array}$$

Encoding special $M_{\triangleleft}(k+1)$:

$$\begin{array}{lll}
\text{Special } \triangleleft_1: & \begin{pmatrix} * & 1 & 0^{k+1} & * \\ & & & \triangleleft_1 \end{pmatrix} \to \begin{pmatrix} * & [2]_{(3)} & * \\ & & \triangleleft_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & 0^{k+1} & * \\ & & & \triangleleft_1 \end{pmatrix} \to \begin{pmatrix} * & [2]_{(3)} & * \\ & & \triangleleft_2 \end{pmatrix} \\
\text{Special } \triangleleft_2: & \begin{pmatrix} * & 1 & 0^{k+1} & * \\ & & & \triangleleft_2 \end{pmatrix} \to \begin{pmatrix} * & [1]_{(3)} & * \\ & & \triangleleft_1 \end{pmatrix} \\
& \begin{pmatrix} * & 2 & 0^{k+1} & * \\ & & & \triangleleft_2 \end{pmatrix} \to \begin{pmatrix} * & [1]_{(3)} & * \\ & & \triangleleft_2 \end{pmatrix} \\
\text{Special } \triangleleft_1 \text{ higher level}: & \begin{pmatrix} * & 0 & 0^{k+1} & * \\ & & & \triangleleft_1 \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ & & \triangleleft_1 \end{pmatrix} \\
\text{Special } \triangleleft_2 \text{ higher level}: & \begin{pmatrix} * & 0 & 0^{k+1} & * \\ & & & \triangleleft_2 \end{pmatrix} \mapsto \begin{pmatrix} * & 0^{k+2} & * \\ & & \triangleleft_2 \end{pmatrix}
\end{array}$$

Figure 1.4: Encoding SMART configurations: $\Phi_{k\mapsto k+1}$ (Part 2: special $\mapsto k+1$)

Encode the special configurations of level $k+1$ into their $(k+1)$-encodings (see the proof of Proposition 1.1 for their correct orbit positions), and preserve the special configurations of level $> k+1$. Rewriting words of length $k+4$.

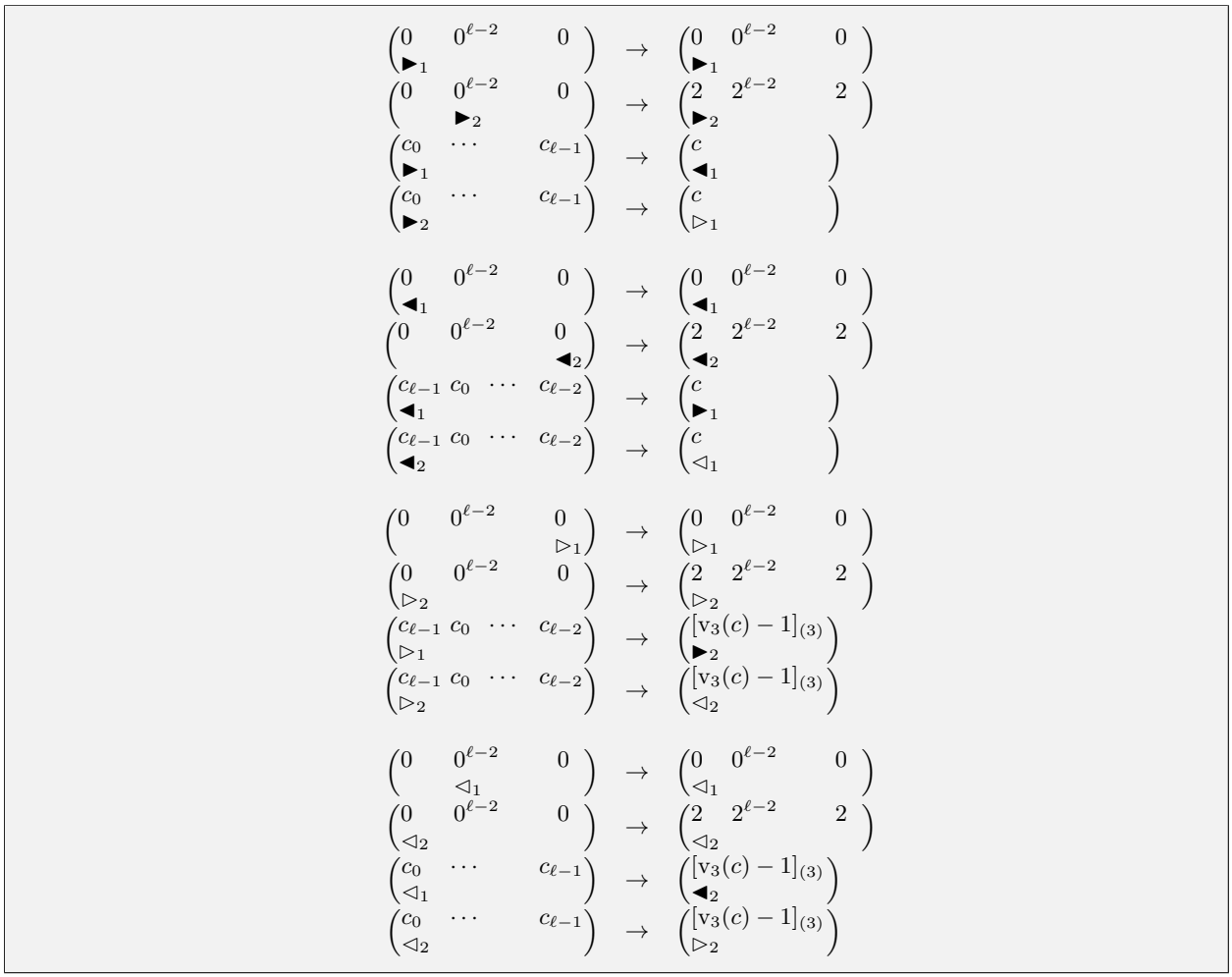


Figure 1.5: Final encoding step of SMART configurations: $\Phi_{\ell,\mathrm{final}} : c_{\ell-1} \mapsto c_*$

This transformation refers directly to the proof of Proposition 1.2, and maps encodings of level $\ell - 1$ to their positions in the orbits of C_q for $q \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\}$. It also "corrects" the position the head and shifts the counter in the encodings of $M_{\blacktriangleleft}(\ell - 1)$ and $M_{\triangleright}(\ell - 1)$, and shifts heads in full-zero configurations whose heads were moved during the encoding of level 0. Rewriting words of length ℓ.

Define Φ_{init} the bijective transformation given in Figure 1.2, $\Phi_{k\mapsto k+1}$ the bijective transformation of Figures 1.3 and 1.4, and $\Phi_{\ell,\text{final}}$ the bijection of Figure 1.5. Given a cyclic configuration c of length ℓ, denote $c_0 = \Phi_{\text{init}}(c)$ its encoding of level 0, $c_{k+1} = \Phi_{k\mapsto k+1}(c_k)$ for $k \leq \ell - 2$, and $c_* = \Phi_{\ell,\text{final}}(c_{\ell-1})$.

Lemma 1.5

Let c be a cyclic configuration of length ℓ. Then $c_* = \Phi(c)$.

Proof. By induction, one sees from Figure 1.1 that c_k is the encoding of level k of c, for $k \leq \ell - 1$. We then conclude that $c_* = \Phi(c)$ by the proof of Proposition 1.2 by comparing the formulas, and recalling the structure of the orbits of $C_{\blacktriangleright}, C_{\blacktriangleleft}, C_{\triangleright}, C_{\triangleleft}$ (to check that we shift in the correct direction when the counter overflows). □

2

Finitary distortion of the SMART machine

This chapter introduces a slightly altered version of $T_{\ell,\mathcal{S}}$ (called the *decorated SMART*) acting on the cyclic tapes of some $C_{\ell,Q_{\mathrm{dec}},\Gamma}$, and establishes Lemma 2.1: this automorphism is "distorted" in some finitely-generated group $G_{\ell,Q_{\mathrm{dec}},\Gamma}$, in the sense that all its powers (including ones exponential in ℓ) have word norm polynomial in ℓ under the fixed generators (which is exponentially lower than the order of the group would suggest).

Section 2.1 introduces some context and states Lemma 2.1: $T_{\ell,\mathcal{S}_{dec}}$ is finitary distorted. In particular, it introduces the finitely generated groups $G_{\ell,Q,\Gamma}$, which contains all Turing machines with states Q and tape-letters Γ. Using ideas from the study of reversible gates[1], it is not difficult to see that every even tape-permutation of Γ^ℓ (relatively to the position of the head) belongs in $G_{\ell,Q,\Gamma}$. However, as the diameter of the alternating group in $G_{\ell,Q,\Gamma}$ is superlinear in ℓ, we cannot conclude that every power of $T_{\ell,\mathcal{S}}$ has small word norm directly on these abstract grounds.

Section 2.2 proves Lemma 2.3 (which is essentially Barrington's theorem [Bar89]) and some applications: it is possible to condition permutations of the head by the remaining $\ell-1$ tape-symbols, i.e. apply the permutation only if some condition holds on the $\ell-1$ tape-letters to its right. Furthermore, this is efficient: if the condition deals with $\ell-1$ bits and belongs in the complexity class NC^1, i.e. it admits an efficient implementation (logarithmic depth in ℓ) by classical gates, then the word norm of the corresponding conditional permutation will be polynomial in ℓ.

Section 2.3 implements the encoding of Section 1.3 in some $G_{\ell,Q_{\mathrm{dec}},\Gamma}$. This essentially proves Lemma 2.1: indeed, this conjugates the application of $(T_{\ell,\mathcal{S}_{\mathrm{dec}}})^n$ to adding the number n to a counter of length ℓ, and we are left with performing such additions in polynomially many steps in ℓ in Lemma 2.10.

2.1 Context and statement of Lemma 2.1

2.1.1 Finitely generated group $G_{\ell,Q,\Gamma}$

Recall that $C_{\ell,Q,\Gamma}$ is the set of finite cyclic tapes of length ℓ (see Section 1.1), with states Q and tape-alphabet Γ, containing at most one head (i.e. a letter in $Q\times\Gamma$).

[1] In this report, *reversible gates* are permutations acting on (a sublanguage of) some A^n that only touch one subset of coordinates at a time, where A is a finite alphabet. See [Sal22, Section 3.3] for more details about reversible permutations, and the result that gates with two inputs are sufficient to generate all the even permutations.

Let $G_{\ell,Q,\Gamma}$ be the finitely generated subgroup of $\mathrm{Sym}(C_{\ell,Q,\Gamma})$ generated by state-dependent moves, and the unary gates permuting the contents of cells containing heads. Formally, for $g \in \mathrm{Sym}(Q \times \Gamma)$, define the *unary gate* $\pi_g \in \mathrm{Sym}(C_{\ell,Q,\Gamma})$ as

$$\pi_g(w)_j = \begin{cases} w_j & \text{if } w_j \in \Gamma \\ g(w_j) & \text{if } w_j \in (Q \times \Gamma) \end{cases}$$

and for $q \in Q$ the *state-dependent (right) move* $\rho_q \in \mathrm{Sym}(C_{\ell,Q,\Gamma})$ as:

$$\rho_q(w)_j = \begin{cases} \pi_\Gamma(w_j) & \text{if } w_{j-1 \bmod \ell} \notin (\{q\} \times \Gamma) \wedge w_j \notin ((Q \setminus \{q\}) \times \Gamma) \\ w_j & \text{if } w_j \in ((Q \setminus \{q\}) \times \Gamma) \\ (q, \pi_\Gamma(w_j)) & \text{if } w_{j-1 \bmod \ell} \in (\{q\} \times \Gamma) \end{cases}$$

Then we define:

$$G_{\ell,Q,\Gamma} = \langle \{\pi_g : g \in \mathrm{Sym}(Q \times \Gamma)\} \cup \{\rho_q \mid q \in Q\} \rangle$$

We can see the group $G_{\ell,Q,\Gamma}$ as the group generated by the instructions of Turing-machines: moving heads based on their states, or permuting their values. To ease notations, we denote $\prod_{q \in Q} \rho_q$ by ρ.

It is easy to see that for any reversible Turing machine $\mathcal{M}$, $T_{\ell,\mathcal{M}}$ is an element of $G_{\ell,Q,\Gamma}$. Indeed, a step of computation is the composition of a head permutation α of $Q \times \Gamma$, followed with state-dependent moves β_{+1} and β_{-1}:

$$\alpha(q,a) = \begin{cases} (q', b) & \text{if } (q, a, q', b) \in \Delta \\ (q', a) & \text{if } (q, \pm 1, q') \in \Delta \end{cases}$$

$$\beta_{+1} = \prod_{q' \mid \exists q, (q,+1,q') \in \Delta} \rho_{q'}$$

$$\beta_{-1} = \prod_{q' \mid \exists q, (q,-1,q') \in \Delta} {\rho_{q'}}^{-1}$$

We denote by $\delta(\ell, n)$ the word norm of $(T_{\ell,\mathcal{M}})^n$ in $G_{\ell,Q,\Gamma}$.

Remark. *It can be shown that $G_{\ell,Q,\Gamma}$ is, for large enough ℓ, $|Q|$ and $|\Gamma|$ (in particular for all versions of the SMART machine we consider and for $\ell \geq 2$), of finite index in the automorphism group of $C_{\ell,Q,\Gamma}$ under the shift action of $\mathbb{Z}_n$. This is not particularly useful, however, as what we need it for is to provide a group where the SMART machine corresponds to an element of small radius (far smaller than the radius of the group).*

2.1.2 Decorated SMART machine

Let $\mathcal{S} = (Q, \Gamma, \Delta)$ be the SMART machine (see Section 1.2). Define the *decorated version* of the SMART machine as $\mathcal{S}_{\mathrm{dec}} = (Q_{\mathrm{dec}}, \Gamma, \Delta_{\mathrm{dec}})$, with

$$\begin{aligned} Q_{\mathrm{dec}} &= Q \times D \times G \\ \Delta_{\mathrm{dec}} &= \bigcup_{(d,x) \in D \times G} \left\{ \Big((q,d,x), a, (q',d,x), b \Big) : (q,a,q',b) \in \Delta \right\} \\ &\quad \cup \bigcup_{(d,x) \in D \times G} \left\{ \Big((q,d,x), \delta, (q',d,x) \Big) : (q,\delta,q') \in \Delta \right\} \end{aligned}$$

for $D = \{d_1, d_2\}$ and $G = [\![0,5]\!] \times \Gamma$, i.e. the states of Q_{dec} now carry a state $q \in Q$ of the original machine $\mathcal{S}$, a special symbol $d \in D$ called the *duck*, and a *ghost symbol* $x \in G$. We have $|Q_{\mathrm{dec}}| = 288$.

The decorated SMART machine $\mathcal{S}_{\text{dec}}$ behaves exactly like $\mathcal{S}$, in the sense that it ignores the new decorations. While the duck component D and ghost component G are completely ignored, they will be used by the generators we use to build powers of the machine efficiently. The point of the *hex component* $[\![0, 5]\!]$ in the ghost is to allow us to condition the application of gates, and to build the permutations we perform in Section 2.2. The *transport component* Γ in the ghost allows us to carry letters around. The duck will be important during intermediate steps of computation in Section 2.2, in order to realize piecewise defined functions.

2.1.3 Main result: Lemma 2.1

Recall that $m : \mathbb{N} \mapsto \mathbb{N}$ is the *movement function*, i.e. $m(n)$ is the maximal number of cells the machine S_{dec} can visit in n steps; and that $\delta(\ell, n)$ is the word norm of $(T_{\ell,\mathcal{S}_{\text{dec}}})^n$ in $G_{\ell,Q_{\text{dec}},\Gamma}$.

Lemma 2.1

Let $\mathcal{S}_{\text{dec}}$ be the decorated version of SMART introduced above:

1. $T_{\mathcal{S}_{\text{dec}}}$ has infinite order.
2. There exist some $C, C' > 0$ such that $m(n) \leq C \log n + C'$.
3. There exists some $d > 0$ such that $\delta(\ell, n) = O(\ell^d)$.

In fact, for $\mathcal{S}_{\text{dec}}$, we can take $C = \ln(2)/\ln(3)$ and $d = 4$.

Any finite order T satisfies the latter two items, and any non-trivial state-dependent shift satisfies the first and the third items. Achieving the first two items is already difficult, and to our knowledge these properties have only been explicitly shown for the SMART machine and the binary SMART machine [CT20]. We expect that the Kari-Ollinger construction in [KO08] can be used to produce more examples of machines satisfying these two properties (at least $m(n) = O(n/\log n)$ follows from general principles for all these machines [GS17]).

The second item is proved with the following computation: after less than 18 steps, the head of SMART is in state $\triangleright_1$ or $\triangleleft_1$ reading a 0 (ignoring the ghost and the duck). Then SMART is either at the left (for $\triangleright_1$) or right (for $\triangleleft_1$) extremity of some word 0^m for some $m \geq \log_3(k) + 2$, or by [COT17, Lemma 4] it builds around this position some pattern in the set C_m for $m \geq \log_3(k) + 2$ (with the notations of [COT17, Lemma 4]). Either way, after this point, k steps of computations cannot read more than $\log_3(k)+2$ different cells. We omit a detailed proof, as the logarithmic speed of SMART is well known.

The proof of the third item is a matter of programming powers of the machine efficiently with primitive reversible operations. The cyclic configurations acted on by SMART form four disjoint cycles, and the overall idea is to turn a configuration into a direct encoding of its position in the cycle (in ternary notation), using the formulas from Section 1.3. This reduces computing powers of SMART to performing ternary additions. The two following sections are dedicated to the proof of this third item.

2.2 Permutation engineering in $C_{\ell,Q_{\mathrm{dec}},\Gamma}$

Let $g \in \mathrm{Sym}(Q_{\mathrm{dec}} \times \Gamma)$ be a permutation, and $C \subseteq (Q_{\mathrm{dec}} \times \Gamma) \times \Gamma^{\ell-1}$ a subset called a *condition*. Define $\pi_{g,C} : C_{\ell,Q_{\mathrm{dec}},\Gamma} \mapsto C_{\ell,Q_{\mathrm{dec}},\Gamma}$ as $\pi_{g,C}(w) = w$ if there is no head in w, and if $w_i \in Q_{\mathrm{dec}} \times \Gamma$, then:

$$\pi_{g,C}(w)_j = \begin{cases} w_j & \text{if } j \neq i \\ g(w_j) & \text{if } j = i \text{ and } w_j \cdot w_{j+1} \dots w_{n-1} \cdot w_0 \dots w_{j-1} \in C \\ w_j & \text{if } j = i \text{ and } w_j \cdot w_{j+1} \dots w_{n-1} \cdot w_0 \dots w_{j-1} \notin C \end{cases}$$

We call $\pi_{g,C}$ the *conditional application of g under condition C*.

We say C is *g-invariant* if $w_j \dots w_{n-1} \cdot w_0 \dots w_{j-1} \in C$ if and only if $g(w_j) \cdot w_{j+1} \dots w_{n-1} \cdot w_0 \dots w_{j-1} \in C$. If C is g-invariant, then $\pi_{g,C}$ is a bijection. We say that C is *ghost-invariant* if there exists $C' \subseteq ((Q \times D) \times \Gamma) \times \Gamma^{\ell-1}$ such that $C = \pi^{-1}(C')$ where $\pi : ((Q \times D \times G) \times \Gamma) \times \Gamma^{\ell-1} \to ((Q \times D) \times \Gamma) \times \Gamma^{\ell-1}$ is the natural projection. Equivalently, it is g-invariant for all permutations $g \in \mathrm{Sym}(G)$, seen as permutations of the G-component of the head.

We say that a head-permutation $g \in \mathrm{Sym}(Q_{\mathrm{dec}} \times \Gamma)$ is *ghost-ignorant* if it factors through the projection that forgets the ghost symbol. In other words, for any choice of $h \in Q_{\mathrm{dec}} \times \Gamma$, w and $g(w)$ carry the same ghost symbol if w has the state-symbol pair h at the head position, and furthermore the ghost symbol does not affect how the other symbols change.

2.2.1 Permutation conditioning

We now prove that if $g \in \mathrm{Alt}(Q_{\mathrm{dec}} \times \Gamma)$ is ghost-ignorant and C is a condition that is both ghost-ignorant and g-invariant, then $\pi_{g,C} \in G_{\ell,Q_{\mathrm{dec}},\Gamma}$. We also provide upper bounds on its word norm depending on C. We begin with what is essentially Barrington's theorem [Bar89].

As a first step, we consider permutations that are the opposite of ghost-ignorant, i.e. only touch the ghost-component G of the head. Let $g \in \mathrm{Alt}(G)$, considered as a subgroup of $\mathrm{Alt}(Q_{\mathrm{dec}} \times \Gamma)$ only permuting the ghost-information in Q_{dec}, and C a ghost-ignorant subset of $(Q_{\mathrm{dec}} \times \Gamma) \times \Gamma^{\ell-1}$. Then note that $\pi_{g,C}$ belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$.

Lemma 2.2

Denote $T : \mathcal{GI} \to \mathbb{N}$ be the optimal function such that $\|\pi_{g,C}\| \leq T(C)$ for all $g \in \mathrm{Alt}(G)$ (considered as a subgroup of $\mathrm{Sym}(Q_{\mathrm{dec}} \times \Gamma)$), where $\mathcal{GI} \subset \mathcal{P}((Q_{\mathrm{dec}} \times \Gamma) \times \Gamma^{\ell-1})$ is the set of all ghost-invariant sets. Then T satisfies the following inequalities:

$$\begin{aligned} T([a]_j) &\leq |\min(j, \ell - j)| \\ T(C \cap C') &\leq 2\Big(T(C) + T(C')\Big) \\ T(C \cup C') &\leq \begin{cases} T(C) + T(C') & \text{if } C \cap C' = \emptyset \\ 2\Big(T(C) + T(C')\Big) + 5 & \text{otherwise} \end{cases} \\ T(C^c) &\leq T(C) + 1 \end{aligned}$$

Proof. We prove by induction over C that every $\pi_{g,C}$, for $g \in \mathrm{Alt}(G)$ (considered as a subgroup of $\mathrm{Sym}(Q_{\mathrm{dec}} \times \Gamma)$), has word norm that checks the aforementioned inequalities.

Case 1. If $C = C' \times \Gamma^{\ell-1}$ for some $C' \subseteq (Q \times D \times G)$ that is ghost-invariant, then any such $\pi_{g,C}$ already appears in the set of generators of $G_{\ell,Q_{\mathrm{dec}},\Gamma}$.

Case 2. If $C = [x]_j$ for some $j \in [\![-\ell, \ell]\!]$, $j \neq 0$ and $x \in \Gamma$, define $C'' = (Q \times D \times G) \times \{x\}$ and $C' = C'' \times \Gamma^{\ell-1}$. Then one can conjugate $\pi_{g,C'}$ (which belongs to $G_{\ell,Q_{\text{dec}},\Gamma}$ by the first item) with ρ^j: the resulting permutation applies g on the ghost symbol if and only if j cells away from the head, the content of the tape is x.

Case 3. If $C = C_1^c$, then $\pi_{g,C_1^c} = \pi_{g^{-1},C_1} \circ \pi_g$.

Case 4. If $C = C_1 \cap C_2$. We use the "commutator trick". As G has cardinality greater than 5, g is a commutator by Ore's theorem [Ore51, Theorem 7]: there exist g_1, g_2 such that $g = [g_1, g_2]$. By the induction hypothesis and a straightforward calculation, we conclude that:

$$\pi_{g,C_1\cap C_2} = \left[\pi_{g_1,C_1}, \pi_{g_2,C_2}\right]$$

Case 5. If $C = C_1 \cup C_2$, with $C_1 \cap C_2 = \emptyset$. Then $\pi_{g,C_1\cup C_2} = \pi_{g,C_1} \circ \pi_{g,C_2}$.

Case 6. If $C = C_1 \cup C_2$, then $\pi_{g,C_1\cup C_2} = \pi_{g,({C_1}^c\cap {C_2}^c)^c}$.

We conclude that $\pi_{g,C} \in G_{\ell,Q_{\text{dec}},\Gamma}$, and that the provided upper-bounds are correct. □

Note that any ghost-ignorant permutation $g \in \text{Sym}(Q_{\text{dec}} \times \Gamma)$ is even since $|G|$ is. Combining this with the previous lemma, we obtain:

Lemma 2.3

Let $T : (Q_{\text{dec}} \times \Gamma) \times \Gamma^{\ell-1} \mapsto \mathbb{N}$ be given by the previous lemma. For any ghost-ignorant permutation $g \in \text{Sym}(Q_{\text{dec}} \times \Gamma)$, and any ghost- and g-invariant condition $C \subseteq (Q_{\text{dec}} \times \Gamma) \times \Gamma^{\ell-1}$, the permutation $\pi_{g,C}$ belongs to $G_{\ell,Q_{\text{dec}},\Gamma}$ with word norm $O(T(C))$.

Proof. Consider $G = [\![0, 5]\!] \times \Gamma$ as $G = \{0, 1\} \times G'$, for $G' = \{0, 1, 2\} \times \Gamma$. As a remark, the structure of G' has no importance in this proof and the reader can consider G' as a black box – in particular Γ never refers to the transport Γ-component of G' in what follows, but instead to a tape letter. For the rest of the proof, fix C some ghost-invariant condition.

Define $H'' = (Q \times D) \times \Gamma$ and $H' = (Q \times D \times \{0, 1\}) \times \Gamma$, considered as projections of $H = (Q \times D \times \{0, 1\} \times G') \times \Gamma = Q_{\text{dec}} \times \Gamma$. Then any permutation $g'' \in \text{Sym}(H'')$ lifts into a permutation $g' = \chi(g'') \in \text{Sym}(H')$ (by ignoring the bit), and any $g' \in \text{Sym}(H')$ lifts into a permutation $g = \xi(g') \in \text{Sym}(H)$ (by ignoring G'). We first prove that the ghost-ignorant permutations $g \in \text{Sym}(H)$ such that C is g-invariant are generated by the $\xi(\sigma') \in \text{Sym}(H)$, for σ' the 3-cycles of $\text{Sym}(H')$ such that C is $\xi(\sigma')$-invariant.

For any ghost-ignorant $g \in \text{Sym}(H)$, there exists some $g'' \in \text{Sym}(H'')$ such that $g = \xi(\chi(g''))$. Additionally, $g'' \in \text{Sym}(H'')$ decomposes into a product of cycles of disjoint support. Without any loss of generality, we can assume that g'' is a cycle of support $S'' \subseteq H''$. So g'' can be considered as a permutation of $\text{Sym}(S'')$. Then g'' lifts into a permutation $g' = \chi(g'') \in \text{Alt}(H')$ that can be considered as a permutation of $\text{Alt}(S')$, for $S' = S'' \times \{0, 1\}$. The group $\text{Alt}(S')$ is generated by its 3-cycles, and any permutation h' of support S' lifts into a permutation $h = \xi(h') \in \text{Sym}(H)$ such that C is h-invariant, because S'' is the support of the cycle g'' and C is g-invariant.

Now, for any 3-cycle $\sigma' \in \text{Sym}(H')$ with C being $\xi(\sigma')$-invariant, we build $\pi_{\xi(\sigma'),C}$. According to the previous paragraph, this will conclude the proof. First, let $r = (0 \cdot z_1, 0 \cdot z_2, 0 \cdot z_3) \in \text{Alt}(G)$ (for some three arbitrary distinct $z_i \in G'$) and pick an arbitrary $x = ((q, d), a) \in S'' \subseteq H''$. Now $\pi_{r,C\cap[x]_0}$ belongs to $G_{\ell,Q_{\text{dec}},\Gamma}$ with word norm $O(T(C))$, where T is defined in the previous lemma. Here we see $\pi_{r,[x]_0}$ as the 3-cycle

$$(((q, d, 0, z_1), a)\ \ ((q, d, 0, z_2), a)\ \ ((q, d, 0, z_3), a))$$

in $\mathrm{Alt}(H)$.

Denote $(h'_1, h'_2, h'_3) = \sigma'$ with $h'_i = ((q_i, d_i, g_i), a_i) \in S' \subseteq H'$. By conjugating $\pi_{r,C\cap[x]_0}$ with the three following permutations of $\mathrm{Sym}(H)$ for $i = 1, 2, 3$:

$$h_i'^{(z)} \leftrightarrow ((q, d, 0, z_i), a)$$

we build in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with word norm $O(T(C))$ the element $\pi_{(h_1'^{(z)},h_2'^{(z)},h_2'^{(z)}),C}$, where $h_i'^{(z)} \in H$ is the lift of h'_i whose second ghost-component is z, i.e. $h_i'^{(z)} = ((q_i, d_i, (g_i, z)), a_i)$. We obtain:

$$\pi_{\sigma,C} = \prod_{z\in G'} \pi_{(h_1'^{(z)},h_2'^{(z)},h_2'^{(z)}),C}$$

□

Readers with a background in complexity theory will find the following version of the statement useful; it is immediate from the definition of NC^1 and the previous lemmas.

Lemma 2.4

Let L be a language in NC^1, and L_n its words of size n. For any ghost-ignorant $g \in \mathrm{Alt}(Q_{\mathrm{dec}} \times \Gamma)$, if L_n is ghost- and g-invariant, then $f_{g,L_n} \in G_{\ell,Q_{\mathrm{dec}},\Gamma}$ has polynomial word norm in n.

We now provide upper bounds on Lemma 2.3 for a few specific conditions, following the proof of Lemma 2.2.

Lemma 2.5

Let $C \subseteq (Q_{\mathrm{dec}} \times \Gamma)$ be a condition of the form $\sim c$, for $c \in (\{0, 1, 2\} \cup \{\bullet\})^j$ with $j \leq \ell$ and $\sim \in \{=, <, \leq, >, \geq\}$ some lexicographic (in)-equality, i.e. $w \in C$ if and only if, if $J \subseteq [\![0, j-1]\!]$ is the ordered set of non-"$\bullet$" indices of c, then $w|_J \sim c|_J$.
Then $T(C) = O(|c|^2)$.

Proof. We use the fact that the permutations of Lemma 2.2 are only permuting the ghost to reduce the head movement. This is a matter of divide and conquer: for $w \in \{0, 1, 2\}^j$ and $(c \in \{0, 1, 2\} \cup \{\bullet\})^j$, if J is the set of non-"$\bullet$" indices of c, then:

$$\begin{aligned} w = c &\iff \left(w_{[\![0,j'-1]\!]\cap J} = c_{[\![0,j'-1]\!]\cap J}\right) \wedge \left(w_{[\![j',j-1]\!]\cap J} = c_{[\![j',j-1]\!]\cap J}\right) \\ w < c &\iff \left(w_{[\![0,j'-1]\!]\cap J} < c_{[\![0,j'-1]\!]\cap J}\right) \\ &\quad \vee \left(\left(w_{[\![0,j'-1]\!]\cap J} = c_{[\![0,j'-1]\!]\cap J}\right) \wedge \left(w_{[\![j',j-1]\!]\cap J} < c_{[\![j',j-1]\!]\cap J}\right)\right) \end{aligned}$$

So if C is the condition $= c$, and $g_1, g_2 \in \mathrm{Alt}(G)$ (considered as a subgroup of $\mathrm{Alt}(Q_{\mathrm{dec}} \times \Gamma)$), then:

$$\pi_{[g_1,g_2],=c} = \left[\pi_{g_1,=c_{[\![0,j'-1]\!]}}, \rho^{-j'} \circ \pi_{g_2,=c_{[\![j',j-1]\!]}} \circ \rho^{j'}\right]$$

Taking $j' = \lfloor k/2 \rfloor$, we obtain $T(C) = O(|c|^2)$. Similarly, if C is the condition $< c$, we obtain $T(C) = O(|c|^2)$. The other cases follow using elementary Boolean algebra. □

2.2.2 Addition

Lemma 2.6

For $k \in \{0,1,2\}^*, |k| \leq \ell$ and $q_d \in Q \times D$, let $s(k,[q_d])$ be the permutation on the cyclic tapes of $C_{\ell,Q_{\mathrm{dec}},\Gamma}$ that does nothing on configurations w containing no head in $[q_d] = \{q\} \times \{d\} \times G$, and adds the ternary number k to the $|k|$ tape-letters to the right of heads in state $[q_d]$, i.e. if $w_{[\![i,i+|k|-1]\!]} = ([q_d], c_0) \cdot c_1 \dots c_{|k|-1}$:

$$s(k,[q_d])(w)_{[\![i,i+|k|-1]\!]} = ([q_d], c'_0) \cdot c'_1 \dots c'_{|k|-1}$$
$$\text{where } \mathrm{v}_3(c'_0 \dots c'_{|k|-1}]) \equiv \mathrm{v}_3(c_0 \dots c_{|k|-1}) + k \bmod 3^{|k|}$$

and $s(k,[q_d])(w)_j = w_j$ if $j \notin [\![i, i+|c|-1]\!]$.
Then $s(k,[q_d])$ has word norm $O(|k|^3)$ in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$.

Proof. Performing addition modulo $3^{|k|}$ reduces to rotating digits and performing carries. Define $r_{[q_d]} \in \mathrm{Alt}(Q_{\mathrm{dec}})$ (considered as a permutation of $\mathrm{Alt}(Q_{\mathrm{dec}} \times \Gamma)$) the tape-invariant permutation that rotates the letter $x \in \Gamma$ at the head (on the tape, not in the transport component) with the 3-cycle $(0,1,2) \in \mathrm{Alt}(\Gamma)$ if the head is in state $[q_d]$. Below, we perform the standard "school algorithm" for addition.

First, move the head $|k|$ letters to the right by applying $\rho^{|k|-1}$. Apply $\pi_{r_{[q_d]}}$ if $k_{|k|-1} = 1$, $\pi_{r^2_{[q_d]}}$ if $k_{|k|-1} = 2$, or the identity if $k_{|k|-1} = 0$. Then, for $j \in [\![1, |k|-2]\!]$, do (with the notations of Lemma 2.5):

1. Move the head to the left with ρ^{-1}.
2. Denoting $k' = k_{[\![|k|-j,|k|-1]\!]}$, apply $\pi_{r_{[q_d]}} \circ \pi_{r_{[q_d]},<\bullet k'}$ if $k_{|k|-j-1} = 1$, $\pi_{r^2_{[q_d]}} \circ \pi_{r_{[q_d]},<\bullet k'}$ if $k_{|k|-j-1} = 2$, and $\pi_{r_{[q_d]},<\bullet k'}$ if $k_{|k|-j-1} = 0$ (in other words, unconditionally add the corresponding digit of k, and perform a carry if the word to the right proves it necessary).

So $s(k,[q_d])$ is the composition of $O(|k|)$ permutations of $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ whose word norms are $O(|k|^2)$, which concludes the proof. □

2.3 Proof of Lemma 2.1

Recall that $\mathcal{S}_{\mathrm{dec}}$ denotes the decorated version of SMART: in this section, we prove that $(T_{\ell,\mathcal{S}_{\mathrm{dec}}})^n$, the action of the n-th power of $\mathcal{S}_{\mathrm{dec}}$ on the cyclic tapes of $C_{\ell,Q_{\mathrm{dec}},\Gamma}$, has word norm $O(\ell^4)$.

To do so, Section 2.3.1 implements the encoding from Section 1.3 in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with word norm $O(\ell^4)$. Section 2.3.2 implements additions on encoded counters with word norm $O(\ell^3)$. Finally, Section 2.3.3 conjugates $(T_{\ell,\mathcal{S}_{\mathrm{dec}}})^n$ to the addition of n on encoded counters, and concludes the proof of Lemma 2.1.

2.3.1 Implementing the encoding with involutions

The "ducking" trick

We indulge in a little digression about a classical idea in reversible computation that we call *ducking*.

Suppose we want to perform a *piecewise-defined permutation* ϕ on a set C: we have finite partitions $\bigsqcup_{i=1}^{p} C_i = C$ and $\bigsqcup_{i=1}^{p} D_i = C$, bijections $\phi_i : C_i \to D_i$, and ϕ can be written as the union $\phi = \bigsqcup_i \phi_i$ (of graphs of functions). Suppose now that for each i,

1. we can "efficiently implement" some element $\psi_i : C \to C$ (i.e. such ψ_i has a small norm with respect to some set of generators) such that $\psi_i|_{C_i} = \phi_i$, and
2. the sets D_i have a simple description.

How can we implement the bijection ϕ efficiently?

To do this, we add some auxiliary symbols to C, and consider instead permutations on $C \times \{d_1, d_2\} \times [\![0, 5]\!]$ (you may recognize the duck and the ghost symbols). We may see ϕ_i and ψ_i as permutations on $C \times \{d_1, d_2\}$ by identifying $C \times \{d_1\} \cong C$, and fixing $C \times \{d_2\}$ pointwise; and then as permutations of $C \times \{d_1, d_2\} \times [\![0, 5]\!]$ by completely ignoring the last symbol.

Observe now that $|\{d_1, d_2\} \times [\![0, 5]\!] \geq 5$ and is even, and consider the permutation π_i that flips the symbol in $\{d_1, d_2\}$ (which we call the *duck*) if and only if the word w in the C-component belongs to D_i. By the previous section, if we take "simple description" to mean the existence of an NC^1 circuit computing membership in D_i, then the element π_i has small word norm.

Now conjugate π_i by ψ_i to get $\chi_i = \psi_i^{-1} \circ \pi_i \circ \psi_i$. It turns out that χ_i exchanges $C_i \times \{d_1\} \times [\![0, 5]\!]$ and $D_i \times \{d_2\} \times [\![0, 5]\!]$, and the correspondence between C_i and D_i is precisely ϕ_i, while on other $C_j \times \{d_1\}$ and $D_j \times \{d_2\}$ it has no effect. One may check this by a case analysis, but the simple way to understand this is that conjugation by ψ_i translates the "preimage side" (duck equal to d_1) forward, and does not act on the "image side" (duck equal to d_2). So, when we conjugate the exchange $D_i \times \{d_1\} \times [\![0, 5]\!] \leftrightarrow D_i \times \{d_2\} \times [\![0, 5]\!]$ by ψ_i, in $D_i \times \{d_1\} \times [\![0, 5]\!]$ we now see ψ_i-elements of $C_i \times \{d_1\} \times [\![0, 5]\!]$, so the exchange becomes the desired $C_i \times \{d_1\} \times [\![0, 5]\!] \leftrightarrow D_i \times \{d_2\} \times [\![0, 5]\!]$.

Now, the χ_i have disjoint supports, so they commute. Composing them in any order, by the assumption on the ϕ_i, we obtain the permutation that exchanges $C \times \{d_1\} \times [\![0, 5]\!]$ and $C \times \{d_2\} \times [\![0, 5]\!]$, and for all i, points that belong in $C_i \times \{d_1\} \times [\![0, 5]\!]$ are moved to $D_i \times \{d_2\} \times [\![0, 5]\!]$ by applying ϕ_i in the C-component (and flipping the duck). Finally, when composing this permutation with an additional flip of the duck, we obtain a map that performs the original map ψ on $C \times \{d_1\} \times [\![0, 5]\!]$, and performs ψ^{-1} on $C \times \{d_2\} \times [\![0, 5]\!]$.

Using ducks to implement the encoding with involutions

In this section, we use this ducking trick and the previous lemmas to implement the inductive encoding of Section 1.3 in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$. More precisely, we use the duck of $D = \{d_1, d_2\}$ to "decompose" each step of the encoding, which are piecewise-defined permutations, into products of transpositions that exchange ducks d_1 and d_2.

For $\Phi_{\dots}$ one of the encoding steps in Figure 1.2, 1.3 and 1.4, or 1.5, let $W \mapsto W'$ be one of its cases (i.e. one line of rewriting in one of the aforementioned figures), with $W, W' \subseteq C_{\ell,Q,\Gamma}$.

Denote by $W_{\mathrm{dec}}, W'_{\mathrm{dec}}$ their decorated versions and $W_{\mathrm{dec}}^{d_1}$ (resp. $W_{\mathrm{dec}}^{d_2}$) the subset of W_{dec} whose heads bear duck d_1 (resp. d_2). Finally, let

$$(w \in W_{\mathrm{dec}}^{d_1}) \leftrightarrow (\Phi_{\dots}(w) \in W'^{d_2}_{\mathrm{dec}})$$

be the involution of $\mathrm{Sym}(C_{\ell,Q_{\mathrm{dec}},\Gamma})$ that swaps words $w \in W_{\mathrm{dec}}^{d_1}$ with words $\Phi_{\dots}(w) \in W'^{d_2}_{\mathrm{dec}}$. More precisely, we rewrite the symbols as indicated in the respective figure, swap the duck and we copy all other symbols unchanged.

Lemma 2.7

For any such case $W \mapsto W'$, the permutation $(w \in W_{\mathrm{dec}}^{d_1}) \leftrightarrow (\Phi_{\dots}(w) \in W'^{d_2}_{\mathrm{dec}})$ belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with norm $O(\ell^3)$.

Proof. For any configuration $w \in C_{\ell,Q_{\mathrm{dec}},\Gamma}$, the condition $w \in W_{\mathrm{dec}}^{d_1}$ checks the state of the head in w, the values of some tape-letters, and a counter being non-zero (in the case of Figure 1.3) or full-zero (in the case of Figure 1.4). By Lemma 2.5, denoting $\pi_{d,W_{\mathrm{dec}}}$ the *ducking* permutation that swaps ducks d_1 and d_2 on the heads of words w if and only if $w \in W_{\mathrm{dec}}$, then $\pi_{d,W_{\mathrm{dec}}}$ belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with norm $O(\ell^2)$.

By conjugating $\pi_{d,W_{\mathrm{dec}}}$ by a sequence of permutations conditioned on the duck being d_2, we can then build $(w \in W_{\mathrm{dec}}^{d_1}) \leftrightarrow (\Phi_{\dots}(w) \in W'^{d_2}_{\mathrm{dec}})$ with norm $O(\ell^3)$ (this $O(\ell^3)$ comes from the addition of some number to a counter).

On an example, consider $W \mapsto W'$ being the following transformation between levels k and $k+1$ (this is the third rewrite in Figure 1.3):

$$\begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_1 & \end{pmatrix} \rightarrow \begin{pmatrix} * & [\mathrm{v}_3(c)+f(n)+1]_{(3)} & * \\ \blacktriangleleft_1 & & & \end{pmatrix}$$

Then the condition W_{dec} is the conjunction of the head being in state $\blacktriangleright_1$, on top of the tape-letter 1, and the $k+1$ tape-letters at its left being non-zero. By Lemma 2.5, $\pi_{d,W_{\mathrm{dec}}}$ belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with norm $O(k^2)$. Then, one can conjugate $\pi_{d,W_{\mathrm{dec}}}$ with the following sequence of permutations, which we all apply *conditioned on the head having duck* d_2:

1. Apply the transposition $(1,0) \in \mathrm{Sym}(\Gamma)$ on the tape-letter under the head to change the letter 1 into 0 if the head has duck d_2 (norm $O(1)$). At the moment, we have the permutation:

$$\begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_1^{d_1} & \end{pmatrix} \leftrightarrow \begin{pmatrix} * & c & 0 & * \\ & & \blacktriangleright_1^{d_2} & \end{pmatrix}$$

2. Move the counter c one step to its right, and move the tape-letter 0 under the head $k+1$ steps to its left. To do so (word norm $O(k)$), just shift each letter of the counter one step to its right while using the transport component of the ghost as a temporary buffer, and move the 0 to the leftmost position (applying all these permutations only if the duck is d_2). At the moment, we have the permutation:

$$\begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_1^{d_1} & \end{pmatrix} \leftrightarrow \begin{pmatrix} * & 0 & c & * \\ & \blacktriangleright_1^{d_2} & & \end{pmatrix}$$

3. Apply $s([f(k)+1]_{(3)}, [\blacktriangleright_1^{d_2}])$, namely the permutation that adds $f(k)+1$ to the ternary number to the right of the head (norm $O(k^3)$). Note that we take $[f(k)+1]_{(3)}$ of length $k+2$. Then move the head one step left (norm $O(1)$) if the duck is d_2. At the moment, we have the permutation:

$$\begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_1^{d_1} & \end{pmatrix} \leftrightarrow \begin{pmatrix} * & [\mathrm{v}_3(c)+f(n)+1]_{(3)} & * \\ \blacktriangleright_1^{d_2} & & & \end{pmatrix}$$

4. Apply the transposition that swaps $\blacktriangleright_1^{d_2}$ and $\blacktriangleleft_1^{d_2}$. We finally obtain:

$$\begin{pmatrix} * & c & 1 & * \\ & & \blacktriangleright_1^{d_1} & \end{pmatrix} \leftrightarrow \begin{pmatrix} * & [\mathrm{v}_3(c)+f(n)+1]_{(3)} & * \\ \blacktriangleleft_1^{d_2} & & & \end{pmatrix} \qquad \square$$

Define $C^{d_1}_{\ell,Q_{\mathrm{dec}},\Gamma}$ and $C^{d_2}_{\ell,Q_{\mathrm{dec}},\Gamma}$ the cyclic tapes whose heads respectively have ducks d_1 and d_2. We complete the "ducking" process and obtain as an immediate corollary:

Lemma 2.8

Define $\Pi_{\mathrm{init}}, \Pi_{n\mapsto n+1}$ and $\Pi_{\ell,\mathrm{final}}$ as:

$$\Pi_{\dots} : (w \in C^{d_1}_{\ell,Q_{\mathrm{dec}},\Gamma}) \leftrightarrow (\Phi_{\dots}(w) \in C^{d_2}_{\ell,Q_{\mathrm{dec}},\Gamma})$$

where $\Phi_{\dots}$ is a transformation defined in Section 1.3. Then $\Pi_{\mathrm{init}}, \Pi_{k\mapsto k+1}$ and $\Pi_{\ell,\mathrm{final}}$ all belong in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with word norm $O(\ell^3)$.

Proof. Each of these transformations $\Phi_{\dots}$ is composed of finitely many cases $W \mapsto W'$ of disjoint support and images, i.e. each $\Pi_{\dots}$ can be written as the product of finitely many $(w \in W^{d_1}_{\mathrm{dec}}) \leftrightarrow (\Phi_{\dots}(w) \in W'^{d_2}_{\mathrm{dec}})$. $\square$

Lemma 2.9

Denoting π_d the involution that swaps ducks d_1 and d_2, let

$$\Pi_\ell = \pi_d \circ \Pi_{\ell,\mathrm{final}} \circ \left(\prod_{k=0}^{\ell-2} \pi_d \circ \Pi_{k\mapsto k+1} \right) \circ \pi_d \circ \Pi_{\mathrm{init}}$$

Then $\Pi_\ell \in G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with norm $O(\ell^4)$. And, *assuming* $w \in C^{d_1}_{\ell,Q_{\mathrm{dec}},\Gamma}$, we have $\Pi_\ell(w) = \Phi_{\mathrm{dec}}(w)$, where Φ_{dec} denotes the decorated version of the encoding described in Section 1.3.

Proof. This is an immediate consequence of Lemma 2.8, the stability of $C^{d_1}_{\ell,Q_{\mathrm{dec}},\Gamma}$ and $C^{d_2}_{\ell,Q_{\mathrm{dec}},\Gamma}$ by every $\pi_d \circ \Pi_{\dots}$, and of the correctness of the encoding proved in Lemma 1.5. $\square$

One should note that, on configurations $w \in C^{d_2}_{\ell,Q_{\mathrm{dec}},\Gamma}$, Π_ℓ "produces garbage": we claim no meaningful interpretation for the image of such w.

2.3.2 Additions on the encoding space

Lemma 2.10

Define $+_{\ell,n,d_1}$ as the bijection of $C_{\ell,Q_{\mathrm{dec}},\Gamma}$ that performs the addition of n in base $\ell \cdot 2 \cdot 3^\ell$ in the sense of our encoding, i.e. $+_{\ell,n,d_1}$ is the shift-commuting bijection defined by $+_{\ell,n,d_1}(w) = w$ if w contains a head with duck d_2 or no head at all, and if $w = ([t], c_0) \cdot c_1 \dots c_{\ell-1}$ with $t \in Q_{\mathrm{dec}}$ has duck d_1 and Q-projection $q_i \in \{\blacktriangleright, \blacktriangleleft, \triangleright, \triangleleft\} \times \{1, 2\}$, then $+_{\ell,n,d_1}(w) = w'$, where:

$$w' = \begin{cases} \sigma^{-j}\left(([t'], c'_0) \cdot c'_1 \dots c'_{\ell-1}\right) & \text{if } q \in \{\blacktriangleright, \triangleright\} \\ \sigma^{j}\left(([t'], c'_0) \cdot c'_1 \dots c'_{\ell-1}\right) & \text{if } q \in \{\blacktriangleleft, \triangleleft\} \end{cases}$$

where t' is equal to t with the Q-component q_i replaced by $q_{i'}$, j is the quotient of $(i-1) \cdot 3^\ell + \mathrm{v}_3(c) + n$ by $2 \cdot 3^\ell$, and $(i'-1) \cdot 3^\ell + \mathrm{v}_3(c')$ is the remainder.
Then $+_{\ell,n,d_1}$ belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with word norm $O(\ell^3)$.

Proof. As rotating the whole tape at most ℓ times can be done with word norm $O(\ell^2)$, we can assume that $0 \leq n < 2 \cdot 3^\ell$.

First, the addition of n modulo $2 \cdot 3^\ell$ (and without rotating the tape if the addition overflows), conditioned on the head having duck d_1, can be implemented with norm $O(\ell^3)$. The proof is very similar to Lemma 2.6, as we consider the configuration w as counters of $\{1, 2\} \cdot \{0, 1, 2\}^\ell$, the only difference being the bit $\{1, 2\}$ being carried by the state q_1 or q_2.

We now need to perform a rotation of the tape if the addition modulo $2 \cdot 3^\ell$ of the previous paragraph overflowed, which is equivalent to $(i'-1) \cdot 3^\ell + \mathrm{v}_3(c') \equiv (i-1) \cdot 3^\ell + \mathrm{v}_3(c) + n \bmod 2 \cdot 3^\ell$ being strictly smaller than n. Define $\pi_{\blacktriangleright,\blacktriangleleft}$ (resp. $\pi_{\triangleright,\triangleleft}$) the involution that exchanges states $\blacktriangleright$ and $\blacktriangleleft$ (resp. $\triangleright$, $\triangleleft$) if $(i'-1) \cdot 3^\ell + \mathrm{v}_3(c') < n$ (and, of course, if they both have duck d_1). By Lemma 2.5, this involution has word norm $O(\ell^2)$.

Then, if $r_\blacktriangleright$ (resp. $r_\triangleright$) rotates the whole tape right if the head is in state $(\blacktriangleright_i, d_1, \cdot)$ (resp. $(\triangleright_i, d_1, \cdot)$ (this permutation belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$: use the transport component of the ghost as a temporary buffer), the commutator $[\pi_{\blacktriangleright,\blacktriangleleft}, r_\blacktriangleright]$ rotates the whole tape right if the state is $\blacktriangleright$ and $(i'-1) \cdot 3^\ell + \mathrm{v}_3(c') < n$, and rotates the whole tape left if the state is $\blacktriangleleft$ and $(i'-1) \cdot 3^\ell + \mathrm{v}_3(c') < n$. Similarly, $[\pi_{\triangleright,\triangleleft}, r_\triangleright]$ rotates $\triangleright$-tapes right and $\triangleleft$-tapes left if $(i'-1) \cdot 3^\ell + \mathrm{v}_3(c') < n$.

We conclude that $+_{\ell,n,d_1}$ (the composition of these steps) indeed belongs in $G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with norm $O(\ell^3)$. □

2.3.3 End of the proof

Lemma 2.11

Define:

$$T^{d_1}_{\ell,\mathcal{S}_{\mathrm{dec}}}(c) = \begin{cases} T_{\ell,\mathcal{S}_{\mathrm{dec}}}(c) & \text{if } c \in C^{d_1}_{\ell,Q_{\mathrm{dec}},\Gamma} \\ c & \text{otherwise} \end{cases}$$

Then $T^{d_1}_{\ell,\mathcal{S}_{\mathrm{dec}}} \in G_{\ell,Q_{\mathrm{dec}},\Gamma}$ with word norm $O(\ell^4)$.

Proof. Let $+_{\ell,n,d_1}$ be given by Lemma 2.10. With Lemma 2.9, one can conjugate $+_{\ell,n,d_1}$ with Π_ℓ and obtain a bijection on $C_{\ell,Q_{\mathrm{dec}},\Gamma}$ that maps configurations of $C^{d_1}_{\ell,Q_{\mathrm{dec}},\Gamma}$ to their n-th iterate by $\mathcal{S}_{\mathrm{dec}}$, and is the identity on $C^{d_2}_{\ell,Q_{\mathrm{dec}},\Gamma}$. In other words:

$$\left(T_{\ell,\mathcal{S}_{\mathrm{dec}}^{d_1}}\right)^n = (+_{\ell,n,d_1})^{\Pi_\ell}$$

Indeed, the addition of $+_{n,k,d_1}$ is only performed on heads having duck d_1. And the shift of the tape, when the addition modulo $2 \cdot 3^\ell$ overflows, is performed to the right (resp. to the left), exactly like $\Phi^{2\cdot 3^\ell}$ performs it on configurations $C_{\blacktriangleright}$ and $C_{\triangleright}$ (resp. $C_{\blacktriangleleft}$ and $C_{\triangleleft}$). □

We can now prove the third point of our main Lemma 2.1 about SMART being distorted on finite cyclic tapes: for $\mathcal{S}_{\mathrm{dec}}$, we have $\delta(\ell, n) = O(\ell^4)$.

Proof. Denoting π_d the "ducking" involution that swaps ducks d_1 and d_2, we have

$$(T_{\ell,\mathcal{S}_{\mathrm{dec}}})^n = \left(\pi_d \circ \left(T_{\ell,\mathcal{S}_{\mathrm{dec}}^{d_1}}\right)^n \circ \pi_d\right) \circ \left(T_{\ell,\mathcal{S}_{\mathrm{dec}}^{d_1}}\right)^n$$

because $(T^{d_1}_{\ell,\mathcal{S}_{\mathrm{dec}}})^{\pi_d} = T^{d_2}_{\ell,\mathcal{S}_{\mathrm{dec}}}$. □

3

Distortion in $\Sigma^{\mathbb{Z}}$ and consequences

Let Σ be a finite alphabet. The set $\Sigma^{\mathbb{Z}}$, once equipped with the prodiscrete topology, becomes a Cantor space called the $\mathbb{Z}$ *full shift*; and a $\mathbb{Z}$ *subshift* (or *subshift* for short if the context is clear) is a closed and shift-invariant subset of $\Sigma^{\mathbb{Z}}$. As mentioned in the general introduction, $\mathbb{Z}$ subshifts are also defined in terms of forbidden patterns.

A recent trend in symbolic dynamics is the study of automorphism groups of subshifts. Typical activities include the study of restrictions that dynamical properties of the subshift put on these groups, and in turn constructing complicated automorphism groups or subgroups thereof. The former activity has been most successful in the low-complexity setting, see [PS22] for a recent account of the state of the art. The latter activity has been most successful on sofic shifts, in particular a lot is known about the finitely-generated subgroups of automorphism groups of full shifts, see [Sal20] for a listing.

In this chapter, we study the group-theoretic notion of distortion – introduced by Gromov [Gro93] – in the context of automorphism groups of subshifts. If G is a finitely-generated group, we say $g \in G$ is a *distortion element*, or *distorted*, if the word norm $|g^n|$ grows sublinearly (with respect to some, or equivalently any, finite generating set). For groups that are not finitely-generated, we say that an element is distorted if it is distorted in some finitely-generated subgroup.

Two basic examples of groups with distortion elements are the Heisenberg group with presentation $\langle a, b \mid [[a, b], a], [[a, b], b]\rangle$, where the element $[a, b]$ has quadratic distortion, meaning we can represent an element of the form $[a, b]^{\Omega(n^2)}$ by composing n generators; and the Baumslag-Solitar group $\mathrm{BS}(1, 2)$ with presentation $\langle a, b \mid b^{-1}ab = a^2\rangle$ where a is easily seen to be exponentially distorted (i.e. the word norm of a^n grows logarithmically).

The previous examples show that distortion elements can appear in nilpotent and metabelian linear groups. It is known that they cannot appear in biautomatic groups, mapping class groups, and the outer automorphism group of the free group [CF06]. See [GK11; LM18; CF06; GL19; CC20; FH06; Pen20; Nav21] for other distortion-related works.

Getting back to automorphism groups, it is an open problem whether the automorphism group of any subshift can contain a distortion element [CFKP18]. It is not known whether the Heisenberg group [KR90] or the Baumslag-Solitar group $\mathrm{BS}(1, 2)$ embed in $\mathrm{Aut}(A^{\mathbb{Z}})$, or indeed in the automorphism group of any subshift. (It is also open whether the additive group of dyadic rationals $\mathbb{Z}[\frac{1}{2}] \leq \mathrm{BS}(1, 2)$ embeds in $\mathrm{Aut}(A^{\mathbb{Z}})$ [BLR88].)

Besides being an interesting group-theoretic notion, the quest for distortion elements in automorphism groups of subshifts is motivated by several purely symbolic dynamical considerations. First, [CK16, Theorem 1.2] shows that finitely-generated torsion-free subgroups of the automorphism group of a subshift

of polynomial complexity are virtually nilpotent. See [DDMP16, Theorem 5.5] for a similar conclusion for inverse limits of bounded step nilsystems. If we could rule out distortion in such examples, we could conclude virtual abelianness.

Second, it is known that the Baumslag-Solitar group, more generally any group with an exponentially distorted element, does not embed in the automorphism group of a zero-entropy subshift [CFKP18]. More precisely, it was observed there that the Morse-Hedlund theorem allows one to translate a distortion element into a lower bound on the complexity of a subshift. This is notable, as this is the only known restriction for automorphism groups of general zero-entropy subshifts. Thus, distortion looks like a natural candidate for restrictions on automorphism groups of general subshifts (as far as the authors know, no restrictions are known on countable subgroups).

Third, distortion is tied to an intrinsic notion in automorphism group theory, namely the growth of the *radius* (a.k.a. range) of the automorphism, when seen as a cellular automaton. Namely, distortion in the group sense implies sublinear growth of the radius [CFK19]. It is not immediately obvious that even sublinear radius growth is possible (indeed this was left open in [CFK19]), but several examples of sublinear radius growth have been constructed. The most relevant for us is the observation from [GS17] that one can even obtain sublinear radius growth in the automorphism group of a full shift: the so-called *SMART machine*, when simulated by an automorphism, gives rise to such growth.

While distortion elements have not previously been exhibited in automorphism groups of subshifts, some facts are known about their dynamics (mostly related the notion of radius). Links to expansive directions and Lyapunov exponents are shown in [CFK19]. A related result is shown in [BDM22], namely distortion elements of automorphism groups of general expansive systems can not themselves be expansive. Links to the dimension group action and inertness are discussed in [CFK19; Sch20].

This chapter proves Theorem 3.4, which is the main result of this intership: the automorphism group of every non-trivial full-shift contains a distortion element of infinite order.

Section 3.1 recalls some useful definitions about subshifts, cellular automata, automorphisms and group distortion.

Section 3.2 details the conveyor belt trick, which we use to embed Turing machines into non-trivial full-shifts. It also precisely states Theorem 3.4 about distortion in every non-trivial full-shift, and provides the necessary mathematical context around the intermediary results.

Sections 3.3 and 3.4.1 focus on the proof of the general distortion result Lemma 3.5, which states that any machine that satisfies the properties from Lemma 2.1 gives rise to a distorted automorphism on a full shift. Section 3.4.2 concludes the proof of Theorem 3.4 with Lemma 3.9, which shows how to optimize the degree to four in the case of the SMART machine.

Sections 3.5 lists some interesting consequences of Theorem 3.4. Surprisingly, some of these consequences reach further than automorphism groups of subshifts: for example, Theorem 3.4 implies the existence of distortion elements of infinite order in the multi-dimensional Brin-Thompson groups mV ($m \geq 2$) by Theorem 3.18, which attracted the interest of mathematicians studying group geometry.

Finally, Section 3.6 considers a few open follow-up questions.

3.1 Definitions

3.1.1 Subshifts

Let Σ be a finite alphabet. An element $x \in \Sigma^{\mathbb{Z}}$ is called a *configuration*. An element $w \in \Sigma^*$ is called a *word* or a *pattern*, and a pattern $w \in \Sigma^*$ is said to *appear* in a configuration $x \in \Sigma^{\mathbb{Z}}$, denoted $w \sqsubseteq x$, if there exists some $i \in \mathbb{Z}$ such that $x_{i+j} = w_j$ for every $j \in [\![0, \mathrm{len}(w) - 1]\!]$.

In the following, Σ is equipped with the discrete topology and $\Sigma^{\mathbb{Z}}$ with the product topology. The latter is generated by the cylinders $[a]_j = \{x \in \Sigma^{\mathbb{Z}} : x_j = a\}$ for $a \in \Sigma$ and $j \in \mathbb{Z}$. This makes $\Sigma^{\mathbb{Z}}$ a Cantor space. The *left shift* $\sigma : \Sigma^{\mathbb{Z}} \mapsto \Sigma^{\mathbb{Z}}$ defined by $\sigma(x)_i = x_{i+1}$ is a $\mathbb{Z}$ action on $\Sigma^{\mathbb{Z}}$. For a more detailed introduction, we refer the reader to [LM95, Chapter 1, 2, 3, 6].

Definition 3.1 $\mathbb{Z}$ subshifts

A *subshift* X is a closed and σ-invariant subset of $\Sigma^{\mathbb{Z}}$. Equivalently, there exists a (potentially infinite) family of forbidden patterns $\mathcal{F}$ such that:

$$X = X_{\mathcal{F}} \triangleq \left\{c \in \Sigma^{\mathbb{Z}} : \forall w \in \mathcal{F}, w \not\sqsubseteq c\right\}$$

Note that several families of forbidden patterns may define the same subshift.

For X a subshift and $n \in \mathbb{N}$, we denote $\mathcal{L}_n(X)$ the set of finite words of length n that appear in X, and $\mathcal{L}(X) = \bigcup_{n\in\mathbb{N}} \mathcal{L}_n(X)$ its *language*. We say that a subshift X is *sofic* if $\mathcal{L}(X)$ is a regular language.

3.1.2 Automorphisms

In this section, we briefly introduce the automorphism group of a subshift.

Definition 3.2 Morphisms and automorphisms

Given a subshift X, a continuous and shift-commuting map $f : X \mapsto X$ is called a *morphism* of X. An *automorphism* is a shift-commuting heomeomorphism, i.e. a bijective morphism.

Given a subshift X, the set of its automorphisms is a group under composition. We denote it $\mathrm{Aut}(X)$.

Automorphisms are better understood in terms of reversible block maps (or reversible cellular automata). Given a subshift X, a map $f : X \mapsto X$ is a *cellular automata* if there exists a finite neighborhood $N \subseteq \mathbb{Z}$ and some local function $h : N \mapsto \Sigma$ such that for any $x \in X$, $f(x)_i = h(x_{|i+N})$. Then, by [Hed69, Theorems 3.4 and 5.14], the morphisms of X are exactly its cellular automata; and its automorphisms are exactly its reversible cellular automata, i.e. the bijective maps $f : X \mapsto X$ such that both f and f^{-1} are cellular automata.

3.1.3 Group distortion

Definition 3.3 Group distortion

Given a group G, an element $g \in G$ of infinite order is *(group) distorted* if there exists a finite set $S \subseteq G$ such that

$$|g^n|_S = o(n)$$

where $|g|_S$ is the *word norm* of g in the finitely generated subgroup $\langle S\rangle$, ie. the length of the shortest presentation of g by elements of S.

We can qualify distortion depending on how the distance between g^n and the identity grows: for example, an element g is *exponentially distorted* if it has logarithmic growth, i.e. $|g^n|_S = O(\log n)$. As there is no convention for polylogarithmic growth, the rest of this chapter works directly with word norms rather than with distortion functions[1].

[1]Note that for well-behaved functions, the word norm growth is just the inverse of the distortion function.

3.2 Context and statement of the main result on $(\Sigma_*)^{\mathbb{Z}}$

3.2.1 Main result

The main result of this chapter, and in fact the most important of this report, is that the automorphism group of some full shift (thus any full shift, as the automorphism groups of full shifts with different non-trivial alphabets embed in each other by [KR90]), contains a distortion element of infinite order:

Theorem 3.4

For any non-trivial alphabet Σ, the group $\mathrm{Aut}(\Sigma^{\mathbb{Z}})$ has an element g of infinite order such that $|g^n|_F = O((\log n)^4)$ for some finite set F.

This solves the second subquestion of [CFKP18, Question 5.1] in the affirmative. The element g in this theorem is essentially the SMART machine [COT17], which morally confirms a conjecture of [GS17], although the embedding we use is slightly more involved than the specific one considered in [GS17]. The group $\langle F \rangle$ we use in the proof is detailed in Lemma 3.9.

Most of the present chapter deals with the proof of this theorem, using technicalities from Chapters 1 and 2. In this section, as Turing machines act on the sofic subshifts of bi-infinite tapes containing at most one head, we detail the conveyor belt trick which makes Turing machine act on a full-shift by wrapping heads inside conveyor belts. We then state Lemma 3.5. Together with Lemma 2.1, along with some additional considerations from Lemma 3.9, they prove Theorem 3.4 completely.

3.2.2 The conveyor belt trick

Recall that a Turing machine $\mathcal{M} = (Q, \Gamma, \Delta)$ acts on the subshift $X_{Q,\Gamma}$ (see Section 1.1), i.e. on the set of bi-infinite tapes containing at most one head (i.e. one letter of $Q \times \Gamma$). This set is a sofic subshift.

To make a Turing machine $\mathcal{M} = (Q, \Gamma, \Delta)$ act on a full shift instead, we use the conveyor belt trick (see for example [GS17, Lemma 3]). Let

$$\Sigma_{Q,\Gamma} = \Big(\Gamma^2 \times \{+1, -1\}\Big) \sqcup \Big((Q \times \Gamma) \times \Gamma\Big) \sqcup \Big(\Gamma \times (Q \times \Gamma)\Big)$$

where we call *conveyor bits* the bits of $\{+1, -1\}$ in $(\Gamma^2 \times \{+1, -1\})$, and define the action of $\mathcal{M}$ on $(\Sigma_{Q,\Gamma})^{\mathbb{Z}}$ as the following automorphism $f_{\mathcal{M}}$.

First, any $x \in (\Sigma_{Q,\Gamma})^{\mathbb{Z}}$ uniquely splits into $x = \dots w_{-2}w_{-1}w_0w_1w_2 \dots$ such that for every $i \in \mathbb{Z}$, we have either

$$w_i \in \Big(\Gamma^2 \times \{+1\}\Big)^* \Big(\Big((Q \times \Gamma) \times \Gamma\Big) \cup \Big(\Gamma \times (Q \times \Gamma)\Big)\Big) \Big(\Gamma^2 \times \{-1\}\Big)^*$$
$$\text{or } w_i \in \Big(\Gamma^2 \times \{+1\}\Big)^+ \Big(\Gamma^2 \times \{-1\}\Big)^+$$

with the exception that on some configurations, their might exist a leftmost or rightmost word with an infinite number of $+1$ or -1.[2] We describe the action of $f_{\mathcal{M}}$ on such finite words w: as configurations made of infinitely many finite w_i are dense, and $f_{\mathcal{M}}$ will be uniformly continuous on these, $f_{\mathcal{M}}$ will uniquely extend to an automorphism on the full shift (and this extension is exactly what one expects it to be).

- On words $w \in \Big(\Gamma^2 \times \{+1\}\Big)^+ \Big(\Gamma^2 \times \{-1\}\Big)^+$, we do nothing.

[2] The corresponding decomposition claim in [GS17] has a mistake, as it uses Kleene stars also in the second form.

- On words $w \in \Big(\Gamma^2 \times \{+1\}\Big)^* \Big(\big((Q \times \Gamma) \times \Gamma\big) \cup \big(\Gamma \times (Q \times \Gamma)\big)\Big) \Big(\Gamma^2 \times \{-1\}\Big)^*$, let

$$w' \in \Big(\Gamma^2\Big)^* \Big(\big((Q \times \Gamma) \times \Gamma\big) \cup \big(\Gamma \times (Q \times \Gamma)\big)\Big) \Big(\Gamma^2\Big)^*$$

be the word obtained by erasing the conveyor bits $+1$ and -1 from w. We see w' as a conveyor belt of length $2|w|$, that is the superimposition of a top word u and a bottom word v, glued together at their borders as if the words were laid down on a conveyor belt.

More precisely, let $u = \pi_1(w')$ and $v = \overline{\pi_2(w')}$ the reversal of $\pi_2(w')$. Then one of these words is in Γ^+, and the other is in $\Gamma^*(Q \times \Gamma)\Gamma^*$. Then we make $\mathcal{M}$ act on $(uv)^{\mathbb{Z}}$ (despite it having infinitely many heads, it should be clear what this means, as all the heads move with the same transition), i.e. define

$$u'v' = T_{\mathcal{M}}\left((uv)^{\mathbb{Z}}\right)_{[0,2|w|-1]}$$

Note that $u'v'$ also contains exactly one head. We then rewrap $u'v'$ into a conveyor belt of $(\Gamma^2)^*(((Q \times \Gamma) \times \Gamma) \cup (\Gamma \times (Q \times \Gamma)))(\Gamma^2)^*$, and add conveyor bits $+1$ (resp. -1) to the cell symbols in Γ^2 to the left of the head (resp. right).

This defines how $f_{\mathcal{M}}$ acts on such words w. This can be summarized as: $f_{\mathcal{M}}$ considers such words as a cyclic tape folded in the shape of a conveyor belt, and acts on the cyclic tape.

Note that x and $f_{\mathcal{M}}(x)$ have the same decomposition into a product of conveyor belts, and that if $\mathcal{M}$ is reversible, then $f_{\mathcal{M}}$ is an automorphism of ${\Sigma_{Q,\Gamma}}^{\mathbb{Z}}$.

Another way to see Turing machines in conveyor belts is the following. For $g \in \mathrm{Alt}(Q \times \Gamma)$, define f_g^{up}, f_g^{down} and f_g as the following automorphisms of $\mathrm{Aut}({\Sigma_{Q,\Gamma}}^{\mathbb{Z}})$:

$$f_g^{\mathrm{up}}(x)_i = \begin{cases} x_i & \text{if } x_i \in \Gamma^2 \times \{+1,-1\} \text{ or } x_i \in (\Gamma \times (Q \times \Gamma)) \\ (g(q,a),b) & \text{if } x_i = ((q,a),b) \in ((Q \times \Gamma) \times \Gamma) \end{cases}$$

$$f_g^{\mathrm{down}}(x)_i = \begin{cases} x_i & \text{if } x_i \in \Gamma^2 \times \{+1,-1\} \text{ or } x_i \in ((Q \times \Gamma) \times \Gamma) \\ (b,g(q,a)) & \text{if } x_i = (b,(q,a)) \in (\Gamma \times (Q \times \Gamma)) \end{cases}$$

$$f_g(x)_i = \begin{cases} x_i & \text{if } x_i \in \Gamma^2 \times \{+1,-1\} \\ (g(q,a),b) & \text{if } x_i = ((q,a),b) \in ((Q \times \Gamma) \times \Gamma) \\ (b,g(q,a)) & \text{if } x_i = (b,(q,a)) \in (\Gamma \times (Q \times \Gamma)) \end{cases}$$

For every $q \in Q$, define ρ_q as the right movement of heads in state q *inside their respective conveyor belts*, and $\rho = \prod_{q \in Q} \rho_q$. And define $G_{Q,\Gamma}$ the group they generate:

$$G_{Q,\Gamma} = \langle \{f_{g,\mathrm{up}}, f_{g,\mathrm{down}}, f_g \mid g \in \mathrm{Sym}(Q \times \Gamma)\} \cup \{\rho_q \mid q \in Q\} \rangle$$

The generators of $G_{\ell,Q,\Gamma}$ introduced in Chapter 2 are in direct correspondence with the generators $\{f_g \mid g \in \mathrm{Sym}(Q \times \Gamma)\}$ and $\{\rho_q \mid q \in Q\}$ of $G_{Q,\Gamma}$, and the latter can also be seen as the basic instructions of Turing machines: moving heads based on their states, or permuting their values.

In a similar way as for the cyclic tapes in Chapter 2, it is easy to see that for any machine $\mathcal{M} = (Q, \Gamma, \Delta)$, $f_{\mathcal{M}}$ is defined as the composition $\beta_{-1} \circ \beta_{+1} \circ f_\alpha$:

$$\alpha(q,a) = \begin{cases} (q',b) & \text{if } (q,a,q',b) \in \Delta \\ (q',a) & \text{if } (q,\pm 1,q') \in \Delta \end{cases}$$

$$\beta_{+1} = \prod_{q' \mid \exists q, (q,+1,q') \in \Delta} \rho_{q'}$$

$$\beta_{-1} = \prod_{q' \mid \exists q, (q,-1,q') \in \Delta} {\rho_{q'}}^{-1}$$

3.2.3 Technical result: Lemma 3.5

For an arbitrary Turing machine $\mathcal{M} = (Q, \Gamma, \Delta)$, define the symmetrized Turing machine $\mathcal{M}_s = (Q_s, \Gamma_s, \Delta_s)$, with $Q_s = Q \times D$ where $D = \{d_\rightarrow, d_\leftarrow\}$, with the same tape alphabet Γ, and transitions $\Delta_s = \Delta_{d_\rightarrow} \times (\Delta^{-1})_{d_\leftarrow}$, i.e. the machine whose heads carry ducks $D = \{d_\rightarrow, d_\leftarrow\}$, and acts forward in time on heads having duck $d_\rightarrow$ and backward in time on heads with duck $d_\leftarrow$.

Define

$$\begin{aligned}
\Gamma_* &= \Gamma \times \{0,1\} \\
Q_* &= Q_s \times \{+1,-1\} \times \{0,1\}^2 \\
\Sigma_* &= \Sigma_{Q_*,\Gamma_*} = \left(\Gamma_*^2 \times \{+1,-1\}\right) \sqcup \left((Q_* \times \Gamma_*) \times \Gamma_*\right) \sqcup \left(\Gamma_* \times (Q_* \times \Gamma_*)\right)
\end{aligned}$$

where the bit of $\{0,1\}$ in Γ_* is the *tape-ghost*, and the value $\{+1,-1\}\times\{0,1\}^2$ in Q_* is the *state-ghost*. Both are technicalities: the tape-ghosts are used as temporary markings to check the lengths of conveyor belts, and the state-ghosts make permutations even and increase the cardinality of our sets of even permutations to make them perfect groups. Note that the $\{+1,-1\}$ in Σ_* are not ghosts, but *conveyor bits*: they point in the direction of the unique head in the conveyor belt containing the cell, if there is one.

Finally, define θ the right movement of the heads *disregarding the conveyor belt structure*, i.e. for $x \in (\Sigma_*)^{\mathbb{Z}}$, if $\pi_{\text{up}} : \Sigma_* \mapsto \Gamma_*$ returns the top tape-letter, $\pi_{\text{down}} : \Sigma_* \mapsto \Gamma_*$ returns the bottom tape-letter, and $\pi_{\text{sign}} : \Sigma_* \mapsto \{+1,-1\}$ returns the conveyor bit of the head or the tape cell:

$$\theta(x)_i = \begin{cases} \Big(\big((q, \pi_{\text{sign}}(x_i), g_{\text{host}}), \pi_{\text{up}}(x_i)\big), \pi_{\text{down}}(x_i)\Big) & \text{if } x_{i-1} \in (\{(q, \pm 1, g_{\text{host}})\} \times \Gamma) \times \Gamma \\ \Big(\pi_{\text{up}}(x_i), \big((q, \pi_{\text{sign}}(x_i), g_{\text{host}}), \pi_{\text{down}}(x_i)\big)\Big) & \text{if } x_{i-1} \in \Gamma \times (\{(q, \pm 1, g_{\text{host}})\} \times \Gamma) \\ ((\pi_{\text{up}}(x_i), \pi_{\text{down}}(x_i)), \pi_{\text{sign}}(x_i)) & \text{if } x_{i-1} \in \Gamma^2 \times \{+1,-1\} \end{cases}$$

And define

$$G_* = \langle G_{Q_*,\Gamma_*} \cup \{\theta\} \rangle$$

the finitely-generated group generated by θ and the Turing-machine instructions of G_{Q_*,Γ_*}. The automorphism θ is only one used in the proof of Lemma 3.8, and is the only generator that can modify the conveyor-belt structure.

We can now establish the second part of the proof of Theorem 3.4. Considering $f_{\mathcal{M}_s} \in \text{Aut}((\Sigma_{Q_s,\Gamma_s})^{\mathbb{Z}})$ as an element of $\text{Aut}((\Sigma_*)^{\mathbb{Z}})$:

Lemma 3.5

If some Turing machine $\mathcal{M}$ satisfies the three properties of Lemma 2.1, then $(f_{\mathcal{M}_s})^n$ has word norm $O(\log^{d+1} n + \log^2 n)$ in $G_* \leq \text{Aut}(\Sigma_*{}^{\mathbb{Z}})$.

Remark. *Note that the condition in this lemma is about $\mathcal{M}$ (and not $\mathcal{M}_s$) verifying the properties of Lemma 2.1, but the conclusion of this lemma is about $(f_{\mathcal{M}_s})^n$ (and not $(f_{\mathcal{M}})^n$).*

Together with Lemma 2.1, this proves that $(f_{\mathcal{S}_{dec,s}})^n$ has word norm $O((\log n)^5)$. After the proof of this lemma, we provide in Section 3.4.2 some additional tricks to lower the upper bound to $O((\log n)^4)$ and complete the proof of Theorem 3.4.

3.3 Permutation engineering in $(\Sigma_*)^{\mathbb{Z}}$

In order to prove Lemma 3.5, the main idea is to build $f_{\mathcal{M}_s}$ (which acts on conveyor belts) from the automorphisms $T_{\ell,\mathcal{M}}$ (which act on finite cyclic tapes) by applying the right $T_{\ell,\mathcal{M}}$ in all the conveyor belts of corresponding length. This is done in the first part of this section. The second part builds a few specific automorphisms required by the *two-scale trick*, a technicality used in Section 3.4.1.

3.3.1 From C_{ℓ,Q_*,Γ_*} to conveyor belts in $(\Sigma_*)^{\mathbb{Z}}$

Lemma 3.6

Let $g \in \mathrm{Sym}(Q_* \times \Gamma_*)$ be a ghost-ignorant permutation.
Then for any $\ell \in \mathbb{N}$ and $\sim \in \{<, \leq, =, \geq, >\}$, $f^{\mathrm{up}}_{g,\mathrm{len}\sim\ell}$ (resp. $f^{\mathrm{down}}_{g,\mathrm{len}\sim\ell}$, $f_{g,\mathrm{len}\sim\ell}$), the automorphism that applies f^{up}_g (resp. f^{down}_g, f_g) in conveyor belts of length $\sim \ell$, has word norm $O(\ell^2)$.

Proof. Let $g \in \mathrm{Alt}(Q_* \times \Gamma_*)$ be an even permutation that is ignorant of the tape-ghost bit of Γ_* and of one state-ghost bit $\{0,1\} \subseteq Q_*$, in the usual sense that it factors through the projection that forgets these bits. Any ghost-ignorant permutation of $\mathrm{Sym}(Q_* \times \Gamma_*)$ fits this condition. We assume that the second state-ghost is the one ignored. We first prove that, for any $\ell \in \mathbb{N}$, the permutation $f^{\mathrm{up}}_{g,\mathrm{ghost}_0=\mathrm{ghost}_\ell}$ (resp. $f^{\mathrm{down}}_{g,\mathrm{ghost}_0=\mathrm{ghost}_\ell}$), that applies f^{up}_g (resp. f^{down}_g) if the tape-ghost under the head and the tape-ghost ℓ steps to its right have equal bit values (and otherwise does nothing), has word norm $O(\ell)$.

As the alphabet $(Q_s \times \{+1,-1\} \times \{0,1\}) \times \Gamma$ has cardinality greater than five, by Ore's theorem [Ore51, Theorem 7], there exist $g_1, g_2 \in G$ such that $g = [g_1, g_2]$. Define $f^{\mathrm{up}}_{g,\mathrm{ghost}_0=\mathrm{ghost}_\ell=0}$ the permutation that applies f^{up}_g if and only if both the tape-ghost of Γ_* under the head and the tape-ghost of Γ_* that is ℓ steps to its right are equal to 0.

Then, denoting $\pi_{\mathrm{ghost}} \in \mathrm{Alt}(Q_* \times \Gamma_*)$ the involution that exchanges the second (ignored) state-ghost $\{0,1\} \subseteq Q_*$ and the tape-ghost of Γ_* under the head, we have:

$$f^{\mathrm{up}}_{g,\mathrm{ghost}_0=\mathrm{ghost}_\ell=0} = \left[\left(f^{\mathrm{up}}_{g_1,\mathrm{ghost}_h=0}\right)^{f_{\pi_{\mathrm{ghost}}}}, \left(f^{\mathrm{up}}_{g_2,\mathrm{ghost}_h=0}\right)^{\rho^{-\ell} \circ f_{\pi_{\mathrm{ghost}}} \circ \rho^{\ell}} \right]$$

where $f^{\mathrm{up}}_{g,\mathrm{ghost}_h=0}$ applies g on the head if the second ghost bit of Q_* is equal to 0. To see this, observe that $\left(f_{g,\mathrm{ghost}_h=0}\right)^{\rho^{-i} \circ f_{\pi_{\mathrm{ghost}}} \circ \rho^{i}}$ applies g if the tape-ghost at (relative) cell i is 0.

Using a similar trick with the value 1 instead of 0, and composing both, we obtain $f^{\mathrm{up}}_{g,\mathrm{ghost}_0=\mathrm{ghost}_\ell}$ that applies f^{up}_g if and only if both the tape-ghost under the head and the tape-ghost ℓ steps to its right are equal. The same reasoning proves the same statement for $f^{\mathrm{down}}_{g,\mathrm{ghost}_0=\mathrm{ghost}_\ell}$.

For $g \in \mathrm{Alt}(Q_* \times \Gamma_*)$ an even permutation that leaves the ghost bit of Γ_* and one ghost bit of Q_* unchanged, we now prove that $f^{\mathrm{up}}_{g,\mathrm{len}|\ell}$ (resp. $f^{\mathrm{down}}_{g,\mathrm{len}|\ell}$) that applies f^{up}_g (resp. f^{down}_g) inside conveyor belts whose length divides ℓ, and is the identity otherwise, has word norm $O(\ell)$.

Indeed, by Ore's theorem [Ore51, Theorem 7] once again, there exists g_1, g_2 such that $g = [g_1, g_2]$. Now, denoting $r_{\mathrm{ghost}} \in \mathrm{Alt}(Q_* \times \Gamma_*)$ the involution that increases (modulo 2) the tape-ghost of Γ_*, applying $f_{r_{\mathrm{ghost}}}$ once modifies both the ghost under the head and the ghost ℓ steps to its right if and only if the length of the conveyor belt divides ℓ. Using this, we obtain:

$$f^{\mathrm{up}}_{g,\mathrm{len}|\ell} = \left[f^{\mathrm{up}}_{g_1,\mathrm{ghost}_0=\mathrm{ghost}_\ell}, \left(f_{r_{\mathrm{ghost}}} \circ f^{\mathrm{up}}_{g_2,\mathrm{ghost}_0=\mathrm{ghost}_\ell} \circ f_{r_{\mathrm{ghost}}} \right) \right]$$

As any power of g belongs in $\mathrm{Alt}(Q_* \times \Gamma_*)$, by going through the divisors of ℓ in decreasing order, we can build any $f^{\mathrm{up}}_{g,\mathrm{len}=\ell}$ with word norm $O(\ell^2)$.[3] For example, if $\ell = 6$,

$$f_{g,\mathrm{len}=6} = f_{g,\mathrm{len}|1} \circ f_{g^{-1},\mathrm{len}|2} \circ f_{g^{-1},\mathrm{len}|3} \circ f_{g,\mathrm{len}|6}.$$

(Our conveyor belts cannot actually have length 1, so $f_{g,\mathrm{len}|1}$ may be dropped.)

We also build any $f^{\mathrm{up}}_{g,\mathrm{len}\leq\ell}$ with word norm $O(n^2)$ by going through the interval $[\![1, \ell]\!]$ in decreasing order and picking suitable powers of g. In particular, we get $f^{\mathrm{up}}_{g,\mathrm{len}\sim\ell}$ for the relations $\sim \ell$ with $\sim \in \{<, \leq, =\}$. From this, the automorphisms with $\sim \in \{\geq, >\}$ are easy to obtain. □

Remark. *One may view the above proof as an instance of Möbius inversion. If g has order m, take $K = \oplus_{\mathbb{Z}_+} \mathbb{Z}_m$ the commutative ring of infinitely many copies of $\mathbb{Z}_m$. We see K as keeping track of how many times g is applied at each conveyor belt length. Define functions $\iota, \gamma : \mathbb{Z}_+ \to K$ where $\iota(n)$ as the indicator function of n (as an element of K), and $\gamma(n)$ the indicator function of the divisor poset of n. Then $\gamma(n) = \sum_{d|n} \iota(d)$ so by Möbius inversion $\iota(n) = \sum_{d|n} \gamma(d)\mu(d, n)$ where μ is the Möbius function of the divisibility poset; thus $\mu(d, n)$ tells us which power of g we should use for each divisor to get $\iota(n)$. The values of ι are a basis of K, so we can get other conditional applications of g with linear combinations.*

Consider now any $T \in G_{\ell,Q_*,\Gamma_*}$. There exists $T_1, \ldots, T_N \in \{\pi_g \mid g \in \mathrm{Alt}(Q_* \times \Gamma_*)\} \cup \{\rho_q \mid q \in Q_*\}$ such that $T = T_N \circ \cdots \circ T_1$. Then, as each generator π_g with $g \in \mathrm{Alt}(Q_* \times \Gamma_*)$ corresponds to a generator f_g of G_*, T defines an automorphism f_T of $\mathrm{Aut}((\Sigma_*)^{\mathbb{Z}})$. The choice of f_T is not canonical, but we can always take it to be defined by the shortest possible formula, or use the formula we used to define T. By construction, if $T \in G_{\ell,Q_*,\Gamma_*}$, then f_T acts like T on conveyor belts of length ℓ (and produces garbage on conveyor belts of length $\neq \ell$).

We now use the previous lemma to condition f_T so that it acts only in conveyor belts of length ℓ, by "symmetrizing" T.

Lemma 3.7

Assume $T \in G_{\ell,Q,\Gamma}$. Then $f_{T_s,\mathrm{len}=\ell} \in \mathrm{Aut}((\Sigma_*)^{\mathbb{Z}})$, which acts like (the lift of) T on conveyor belts of length ℓ having duck $d_\rightarrow$, like (the lift of) T^{-1} on conveyor belts of length ℓ with duck $d_\leftarrow$, and is the identity otherwise, has word norm $O(\|T\| + \ell^2)$ in G_*.

Remark. *Once again, this lemma requires conditions on T as an element of $G_{\ell,Q,\Gamma}$ (and not G_{ℓ,Q_*,Γ_*}), but its conclusion is about $f_{T_s,\mathrm{len}=\ell}$ in $\mathrm{Aut}((\Sigma_*)^{\mathbb{Z}})$ (and not about $f_{T,\mathrm{len}=\ell}$ in $\mathrm{Aut}(\Sigma_{Q,\Gamma})$).*

Proof. Assume $T \in G_{\ell,Q,\Gamma}$. Then T lifts into an automorphism of G_{ℓ,Q_*,Γ_*}, and $T^{d\rightarrow}$ which acts like T on cyclic tapes with duck $d_\rightarrow$, and is the identity otherwise, is also an automorphism of G_{ℓ,Q_*,Γ_*} with word norm $O(\|T\|)$: indeed, any presentation of T in $G_{\ell,Q,\Gamma}$ lifts into a presentation in G_{ℓ,Q_*,Γ_*}, in which we then restrict every generator to apply only on heads with duck $d_\rightarrow$.

Let $d \in \mathrm{Sym}(Q_* \times \Gamma_*)$ be the involution that swaps ducks $d_\rightarrow$ and $d_\leftarrow$ on the head. By Lemma 3.6, $f_{d,\mathrm{len}=\ell}$ has word norm $O(\ell^2)$ and:

$$f_{T_s,\mathrm{len}=\ell} = f_{d,\mathrm{len}=\ell} \circ \left((f_{T^{d\rightarrow}})^{-1} \circ f_{d,\mathrm{len}=\ell} \circ (f_{T^{d\leftarrow}}) \right)$$ □

[3] The number of divisors satisfies $d(\ell) = o(\ell^\epsilon)$ for any $\epsilon > 0$, so we even get word norm $O(\ell^{1+\epsilon})$ for $f^{\mathrm{up}}_{g,\mathrm{len}=\ell}$.

3.3.2 A few specific automorphisms

Define $f_{d,\to t}$ (resp. $f_{d,t\leftarrow}$ and $f_{d,t\leftrightarrow t}$) the automorphism of $\mathrm{Aut}((\Sigma_*)^{\mathbb{Z}})$ that swaps ducks $d_\to$ and $d_\leftarrow$ on the head if the head is at distance less than t from the left border of its conveyor belt, and at least t from the right border of its conveyor belt (resp. distance at least t from the left border and less than t from the right border, resp. distance at least t from both borders).

Define $f_{\mathrm{cb},\to t}$ (resp. $f_{\mathrm{cb},t\leftarrow}$ and $f_{\mathrm{cb},t\leftrightarrow t}$) the automorphism of $\mathrm{Aut}((\Sigma_*)^{\mathbb{Z}})$ that applies the involution $(+1) \leftrightarrow (-1)$ on the conveyor bits carried by $\Gamma_*^2 \times \{+1,-1\}$ or the sign $\{+1,-1\}$ in Q_*, at distance t to the right of the heads (resp. left of the heads, resp. both left and right of the heads), independently of the conveyor belt structures. Note that, these three automorphisms modify the conveyor-belt structures of the configurations they are applied on.

Lemma 3.8

We have:

1. $f_{d,\to t}$, $f_{d,t\leftarrow}$ and $f_{d,t\leftrightarrow t}$ have word norm $O(t^2)$ in G_*.
2. $f_{cb,\to t}$, $f_{cb,t\leftarrow}$ and $f_{cb,t\leftrightarrow t}$ have word norm $O(t^2)$ in G_*.

Proof. For the first item, for $g \in \mathrm{Alt}(Q_*)$ (considered as a subgroup of $\mathrm{Alt}(Q_* \times \Gamma_*)$), there exists $g_1, g_2 \in \mathrm{Alt}(Q_*)$ such that $g = [g_1, g_2] \in \mathrm{Alt}(Q_*)$. With the commutator trick,

$$[f_{g_j}^{\mathrm{up}}, \rho^{-1} \circ f_{g'_j}^{\mathrm{down}} \circ \rho]$$

applies f_g on heads that are exactly in the top-right corner of a conveyor belt. Similar formulas exist for bottom-right, top-left and bottom-left corners of conveyor belts, so that one can condition any such f_g to be applied on heads that are not in the left, right or both corners of their conveyor belts, with word norm $O(1)$.

Then, using a divide and conquer approach with the commutator trick, one can implement $f_{d,\to t}$, $f_{d,t\leftarrow}$ and $f_{d,t\leftrightarrow t}$ in G_* with word norm $O(t^2)$, as in the proof of Lemma 2.5.

Implementing the second item is a bit more ad-hoc, but very simple: we use the right shift θ that moves heads to their right while disregarding the structure of the conveyor belts. Then, if $g \in \mathrm{Sym}(Q_* \times \Gamma_*)$ is the involution that swaps the sign $\{+1,-1\}$ carried by the head regardless of its state,

$$f_{\mathrm{cb},\to t} = \theta^{-t} \circ f_g \circ \theta^t$$

Similar formulas exist for $f_{\mathrm{cb},\leftarrow t}$ and $f_{\mathrm{cb},t\leftrightarrow t}$. □

3.4 Final proofs

3.4.1 Proof Lemma 3.5

We now prove Lemma 3.5: under the assumptions of Lemma 3.5, the automorphisms $(f_{\mathcal{M}_s})^n$ have word norm $O(\log^{d+1} n + \log^2 n)$ in G_*.

Before going into the precise math, here is an overview of the proof. The main idea consists in building $(f_{\mathcal{M}_s})^n$ from the $(T_{\ell,\mathcal{M}})^n$, which by hypothesis have small word norm $O(\ell^d)$. By Lemma 3.7, we can manage this way conveyor belts of increasingly bigger lengths.

However, we can't compose infinitely many of these operations, so after some point we need to manage all the larger conveyor belts at once. To do so, notice that $m(n)$ – the movement of $\mathcal{M}$ in n steps – is assumed to be $O(\log n)$. As a consequence, a head at distance more than $m(n)$ from the borders of its conveyor belts won't see the borders in question. In such a case, we can then create temporary conveyor belts of size $O(\log n)$, apply the corresponding $(T_{\ell,\mathcal{M}})^n$, and erase the temporary borders: the applied operation coincides with $(f_{\mathcal{M}_s})^n$ in the original large conveyor belt.

Let's formalize these ideas. In particular, generating temporary conveyor belts will not work as expected, which justifies the introduction of the "two-scale trick".

Proof. Fix an integer n, which is the power of $f_{\mathcal{M}_s}$ we want to build. Without any loss of generality, assume that n is even. Indeed, if n is odd, then $n-1$ is even, and $\|(f_{\mathcal{M}_s})^n\| = \|(f_{\mathcal{M}_s})^{n-1}\| + O(1)$. Denote $n = 2n'$. With notations from Lemma 2.1, let $L = C \cdot \log n' + C'$. By hypothesis, every $(T_{\ell,\mathcal{M}})^{n'}$ has word norm $O(\ell^d)$ in G_{ℓ,Q_*,Γ_*}.

First, using Lemma 3.7, we manage all conveyor belts of length $< 12L$ with word norm $O(L \cdot L^d)$, as:

$$(f_{\mathcal{M}_s,\mathrm{len}<12L})^{2n'} = \prod_{\ell=1}^{6L-1} f_{(T_{2\ell,\mathcal{M}_s})^{2n'},\mathrm{len}=2\ell}$$

Then, to manage larger conveyor belts, we use what we call the "*two-scale trick*". In the introductory paragraphs, we mentioned the idea of introducing temporary conveyor belts to move heads that originally belonged in very large conveyor belts, and then remove the temporary borders. However, the difficulty lies in properly removing the temporary conveyor belts once the machine has been applied. To solve this, we will actually use temporary conveyor belts twice, with different sizes (hence the name "two-scale trick"). We give a visual explanation of this trick in Figure 3.1.

Define $L_1 = 4L-2$, $L_2 = 8L-2$, $L_3 = 12L-2$. Note that L_1 (resp. L_2) is the length of a conveyor belt constructed by $f_{\mathrm{cb},L\leftrightarrow L}$ (resp. $f_{\mathrm{cb},2L\leftrightarrow 2L}$). With Lemma 3.8 and Lemma 3.7, define:

$$\lambda_{n'} = \Big(f_{\mathrm{cb},2L\leftrightarrow 2L} \circ \Big(f_{(T_{L_2,\mathcal{M}_s})^{n'},\mathrm{len}=L_2}\Big) \circ f_{\mathrm{cb},L\leftrightarrow L} \circ \Big(f_{(T_{L_1,\mathcal{M}_s})^{n'},\mathrm{len}=L_1}\Big)^{-1} \circ f_{\mathrm{cb},2L\leftrightarrow 2L} \circ \Big(f_{(T_{L_1,\mathcal{M}_s})^{n'},\mathrm{len}=L_1}\Big) \circ f_{\mathrm{cb},L\leftrightarrow L}\Big)^{\cdot}$$

Let f_i denote the composition of the first i automorphisms on this list, i.e. $f_1 = f_{\mathrm{cb},2L\leftrightarrow 2L}, \dots, f_7 = \lambda_{n'}$. The actions of the inverses of the automorphisms f_i are illustrated in the left column in Figure 3.1 (on a certain subset of configurations).

Now denote $d \in \mathrm{Alt}(Q_* \times \Gamma_*)$ the ducking involution, i.e. the permutation that flips ducks $d_{\rightarrow}$ and $d_{\leftarrow}$, and consider:

$$f_{\mathcal{M}_s,2n',3L\leftrightarrow 3L} = (\lambda_{n'})^{-1} \circ (f_{d,3L\leftrightarrow 3L}) \circ (\lambda_{n'})\,.$$

We see from Figure 3.1 that, letting $f = f_{d,3L\leftrightarrow 3L}$ and reading the successive partial conjugations f^{f_i} top-down, $f_{d,3L\leftrightarrow 3L}$ gets conjugated to a map that applies our machine $(\mathcal{M}_s)^{n'}$ twice if it is on a conveyor belt that extends sufficiently both left and right, and flips the duck as a side product.

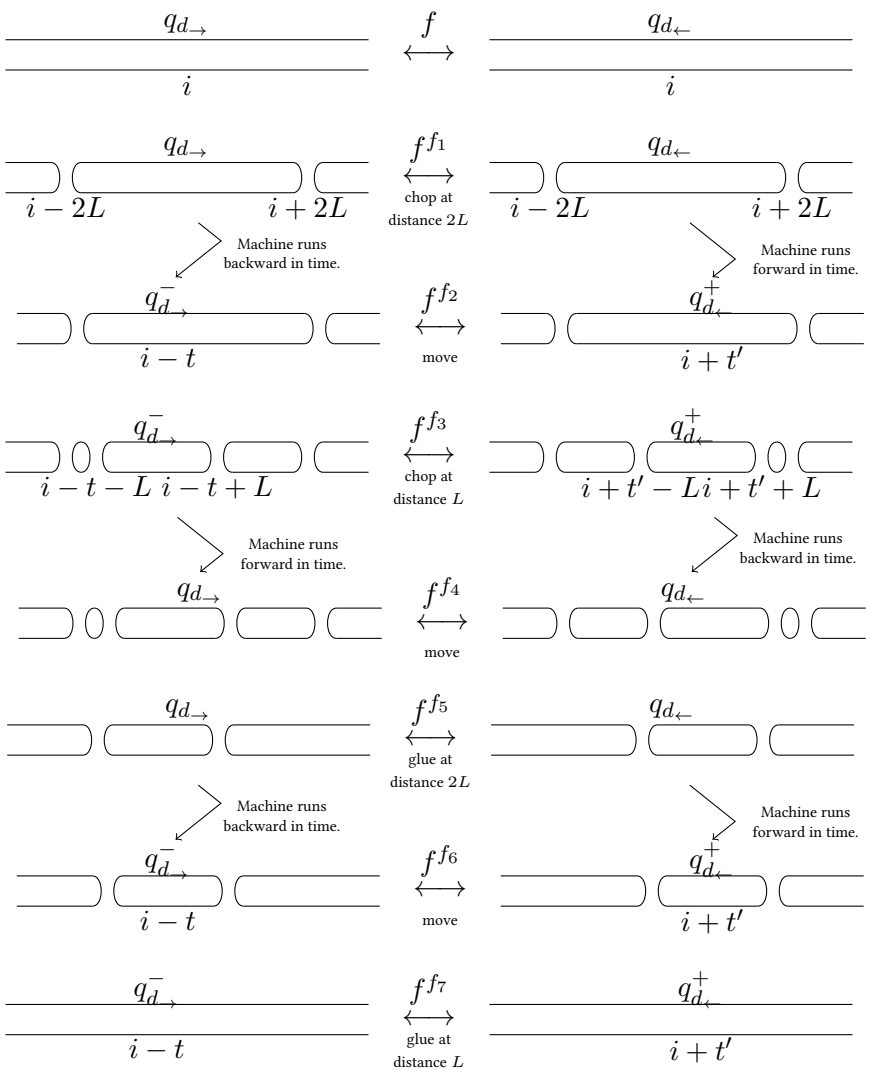


Figure 3.1: The “two-scale trick”

Illustration of the two-scale trick. We show the conveyor belts as lines (without tape letters). The head may be on either track. The head is in position i in state q initially, was in position $i-t$ in state q^- at time $-n'$, and will be in position $i+t'$ in state q^+ at time $+n'$. The automorphism f acts trivially unless we are in the situation of the first line, so the conjugated automorphisms also act nontrivially only in the shown situation. In particular f^{f_7} behaves as expected.

Similar formulas $f_{\mathcal{M}_s,2n',3L\leftarrow}$ (resp. $f_{\mathcal{M}_s,2n',\rightarrow 3L}$) exist, managing heads in large conveyor belts at distance less than $3L$ from their right (resp. left) border. The latter two have word norm $O(L^{d+1}+L^2)$, as we have to replace occurrences of $(f_{(T_{\ell,\mathcal{M}_s})^{n'},\mathrm{len}=\ell})$ with $\prod_{j=1}^{\ell}(f_{(T_{j,\mathcal{M}_s})^{n'},\mathrm{len}=j})$. In any case, these permutations have disjoint support (because the distance to a conveyor belt border is checked "after n' steps of computation") and word norm $O(L^{d+1}+L^2)$.

Now if $d \in \mathrm{Alt}(Q_* \times \Gamma_*)$ denotes the involution that flips ducks $d_\rightarrow$ and $d_\leftarrow$, we have:

$$(f_{\mathcal{M}_s})^n = \left(f_{d,\mathrm{len}\geq L_3} \circ f_{\mathcal{M}_s,n,\rightarrow 3L} \circ f_{\mathcal{M}_s,n,3L\leftarrow} \circ f_{\mathcal{M}_s,n,3L\leftrightarrow 3L}\right) \circ \left(\prod_{n=1}^{6L-1} f_{T_{\ell,\mathcal{M}}{}^n,\mathrm{len}=2l}\right)$$

Which concludes the proof. □

3.4.2 Improving the upper bound on SMART

As the computation of $(T_{\ell,\mathcal{S}_{\mathrm{dec}}})^n$ is "uniform" in ℓ (the k-th step of the encoding, and the k-th step of the addition, are the same on all conveyor belts of length $\geq k$), we can compute the action of SMART on all conveyor belts in parallel. This drops the word norm of $(T_{\mathcal{S}_{\mathrm{dec}},s})^n$ from $O((\log n)^5)$ with Lemma 3.5, to $O((\log n)^4)$ with this new method. We also make minor optimizations to the alphabet, by joining some of the auxiliary symbols used on the Turing machine and automorphism side.

All in all, Lemma 3.9 concludes the proof of Theorem 3.4.

Lemma 3.9

Let $\mathcal{S}$ be the SMART machine introduced in Section 1.2, and define $\mathcal{S}_\star = (\Gamma_\star, Q_\star, \Delta_\star)$ as follows:

$$\begin{aligned}
\Gamma_\star &= \Gamma \times \{0,1\}\\
Q_\star &= Q \times \{d_\rightarrow, d_\leftarrow\} \times \left(\{+1,-1\} \times \{0,1\}^2 \times \Gamma\right)\\
\Delta_\star &= \Big\{(q_\star, a_\star, q'_\star, b_\star), (q_\star, \delta, q'_\star) \mid\\
&\qquad \exists d \in \{d_\rightarrow, d_\leftarrow\}, g \in (\{+1,-1\} \times \{0,1\}^2 \times \Gamma), g' \in \{0,1\}\\
&\qquad\quad q_\star = (q,d,g), q'_\star = (q',d,g), a_\star = (a,g'), b_\star = (b,g'),\\
&\qquad\quad d = d_\rightarrow \implies (q,a,q',b) \in \Delta \text{ or } (q,\delta,q') \in \Delta\\
&\qquad\quad d = d_\leftarrow \implies (q,a,q',b) \in \Delta^{-1} \text{ or } (q,\delta,q') \in \Delta^{-1}\Big\}
\end{aligned}$$

and

$$\Sigma_\star = \Sigma_{Q_\star,\Gamma_\star} = \left(\Gamma_\star^2 \times \{+1,-1\}\right) \sqcup \left((Q_\star \times \Gamma_\star) \times \Gamma_\star\right) \sqcup \left(\Gamma_\star \times (Q_\star \times \Gamma_\star)\right)$$

Then $(f_{\mathcal{S}_\star})^n$ has word norm $O(\log^4 n)$ in $\mathrm{Aut}((\Sigma_\star)^{\mathbb{Z}})$.

Proof. Once again, we prove that $(f_{\mathcal{S}_\star})^n$ has word norm $O(\log^4 n)$ in the following subgroup of $\mathrm{Aut}((\Sigma_\star)^{\mathbb{Z}})$

$$\begin{aligned}
G_{Q_\star,\Gamma_\star} = \langle &\{f_{g,\mathrm{up}}, f_{g,\mathrm{down}}, f_g : g \in \mathrm{Sym}(Q_\star \times \Gamma_\star)\}\\
&\cup \{\rho_q \mid q \in Q_\star\} \cup \{\theta\}\rangle
\end{aligned}$$

We start by explaining the new alphabet. We have dropped the duck $\{d_1, d_2\}$ to fuse it with $\{d_\rightarrow, d_\leftarrow\}$, and we have dropped the hex component $[\![0,5]\!]$ of the ghost. We can use the set $\{+1,-1\} \times \{0,1\}^2$ in

its place, as the only thing we used was that the cardinality is large enough, and that 6 is even. The hex component was also only used temporarily, always returning to its original value after modifications to other components of the state, so it is safe to reuse this set for it.

The reason we have applied the construction without the duck $\{d_1, d_2\}$ is of course that this is the exact same duck; our assumption in the previous section is that we can efficiently apply the Turing machine when the duck is $d_\rightarrow$, while doing nothing on ducks $d_\leftarrow$, and this is exactly how the duck was used in Chapter 2 (see Lemma 2.11).

Now we explain the optimization; we only give a high-level explanation, as this is completely analogous to what was done in the previous section. We only amend the proof above by proving that both $(f_{\mathcal{S}_\star,\text{len}<L_3})^{2n'}$ and $(f_{\mathcal{S}_\star,\text{len}\leq L_3})^{2n'}$ have word norm $O(L^4)$, as they are sufficient to manage both small conveyor belts of length $< L_3$ and large conveyor belts in the two-scale trick.

1. The encoding (word norm $O(L^4)$). Let Π_{init}, $\Pi_{k\mapsto k+1}$ and $\Pi_{\ell,\text{final}}$ be the steps of encoding of $\mathcal{S}$ defined in Lemma 2.8, with $\Pi_{\dots}$ also swapping ducks $d_\rightarrow$ and $d_\leftarrow$:

$$\Pi_{\dots} : (w \in C^{d_\rightarrow}_{\ell,Q_\star,\Gamma_\star}) \leftrightarrow (\Phi_{\dots}(w) \in C^{d_\leftarrow}_{\ell,Q_\star,\Gamma_\star})$$

   As explained above, the generators of $G_{l,Q_\star,\Gamma_\star}$ correspond to generators of $G_{Q_\star,\Gamma_\star}$, so that each $\Pi_{\dots}$ of $G_{l,Q_\star,\Gamma_\star}$ can also be considered as an element $f_{\Pi_{\dots}}$ of $G_{Q_\star,\Gamma_\star}$.

   The key point of this proof consists in understanding that $\Pi_{k\mapsto k+1}$ behaves as expected on every conveyor belt of length $\geq k+2$, and produces garbage on conveyor belts of length $\leq k+1$.

   In other words, let d be the "ducking" involution that swaps ducks $d_\rightarrow$ and $d_\leftarrow$ on heads, and define:

$$\begin{aligned}
f_{\text{init}} &= f_{d,\text{len}<L_3} \circ \left(f^{-1}_{\Pi_{init}} \circ f_{d,\text{len}<L_3} \circ f_{\Pi_{init}}\right)\\
f_{k\mapsto k+1} &= f_{d,k+2\leq\text{len}<L_3} \circ \left(f^{-1}_{\Pi_{k\mapsto k+1}} \circ f_{d,k+2\leq\text{len}<L_3} \circ f_{\Pi_{k\mapsto k+1}}\right)\\
f_{\ell,\text{final}} &= f_{d,\text{len}=l} \circ \left(f^{-1}_{\Pi_{\ell,\text{final}}} \circ f_{d,\text{len}=l} \circ f_{\Pi_{\ell,\text{final}}}\right)
\end{aligned}$$

   Each of these elements has word norm $O(L^2 + L^3)$. Then, define:

$$f_{\text{encode}} = \left(\prod_{\ell=1}^{L3-1} f_{2\ell,final}\right) \circ \left(\prod_{k=1}^{L_3-1} f_{k\mapsto k+1}\right) \circ f_{\text{init}}$$

   The automorphism f_{encode}, which acts on conveyor belts of length $< L_3$ and encodes SMART configurations with ducks $d_\rightarrow$ into their correct encoding, and produces garbage on ducks $d_\leftarrow$, has word norm $O(L^4)$. (A similar automorphism exists for conveyor belts of length $\leq L_3$).

2. The addition of k on ducks $d_\rightarrow$ (word norm $O(L^3)$). We follow the proof of Lemma 2.10. First, we manage addition modulo 3^ℓ in every conveyor belt of length $\ell < L_3 = 12L - 2$. To do so, we use the commutator trick on the proofs of Lemma 2.6 and Lemma 3.6, so that we can build the ternary addition of each of the first ℓ digits of n in base 3 on conveyor belts of length $\leq \ell$ with norm $O(\ell^2)$, which means that adding $n \bmod 3^\ell$ to every conveyor belt of length ℓ for $\ell < L_3$ can be done in word norm $O(L^3)$.

   Let us now focus on addition modulo $2 \cdot 3^\ell$. Let b_ℓ be the integer division of n by 3^ℓ modulo 2, and q_ℓ be the quotient of n by $2 \cdot 3^\ell$. In conveyor belts of length exactly ℓ, we can manage the addition of b_ℓ to the head, while considering the carry of the previous ternary addition, with word norm $O(\ell^2)$. Considering the sizes of conveyor belts one by one, this step can be done on all conveyor belts of length $\ell < L_3$ in word norm $O(L^3)$.

Finally, one has to shift the tape by $q_\ell \bmod \ell$ steps (or $q_\ell + 1 \bmod \ell$ in case of an overflow) left or right (depending on the state) in conveyor belts of size ℓ. Using the same commutator trick on states $\blacktriangleright$, $\blacktriangleleft$ and $\triangleright$, $\triangleleft$ as in the proof of Lemma 2.10, this can be done with word norm $O(\ell^2)$. Considering the sizes of conveyor belts one by one, this step can be done on all conveyor belts of length $\ell < L_3$ in word norm $O(L^3)$.

All in all, the automorphism f_{+n} that performs the addition of n in conveyor belts of all sizes $< L_3$ with ducks $d_\rightarrow$, has word norm $O(L^3)$.

Then, conjugating f_{+n} by f_{encode} performs $(f_{\mathcal{S}_\star})^n$ on heads with duck $d_\rightarrow$ on conveyor belts of length $< L_3$, and is the identity otherwise. Adding the same automorphism conjugated with f_d and composing, we obtain $(f_{\mathcal{S}_\star,\text{len}<L_3})^{2n'}$ with word norm $O(L^4)$. Similar formulas exist for $(f_{\mathcal{S}_\star,\text{len}\leq L_3})^{2n'}$. □

As a final remark, the word norm of this implementation of $(f_{\mathcal{S}_\star})^n$ is $O(\log n \cdot (\omega_\leq(\log n) + \omega_+(\log n))$, where $\omega_\leq(N)$ is the complexity of the lexicographic inequalities on words of length N, and $\omega_+(N)$ is the complexity of the ternary addition on words of length N, both in the setting of reversible gates. While we could not find a way to perform $\omega_+(N)$ with complexity less than $O(N^3)$, it would be interesting to optimize this specific operation by itself.

3.5 Corollaries

We prove several results of interest, which are all straightforward corollaries of Theorem 3.4.

3.5.1 About automorphism groups of subshifts

First, we obtain the characterization of the class of sofic shifts whose automorphism groups have distortion elements.

Theorem 3.10

Let X be a sofic shift. Then $\mathrm{Aut}(X)$ contains a distortion element if and only if X is uncountable.

Proof. If X is uncountable, then $\mathrm{Aut}(A^{\mathbb{Z}}) \leq \mathrm{Aut}(X)$ [Sal18; KR90].

If X is countable, then the proof of Proposition 2 in [ST12] shows that every automorphism $f \in \mathrm{Aut}(X)$ is either periodic or admits a *spaceship*, namely a configuration of the form $x = \ldots uuuuvwww\ldots$ which is not spatially periodic, and $f^n(x) = \sigma^m(x)$ for some $m \neq 0$. Clearly this prevents distortion. □

As another immediate consequence, using the argument of [CFKP18] we obtain that the automorphism group of a full shift cannot be embedded in the automorphism group of a low-complexity subshift. Recall that the *lower entropy dimension* [Mey11] of a subshift is defined by the formula

$$\underline{D}(X) = \liminf_{k\to\infty} \frac{\log(\log N_k(X))}{\log k},$$

where $N_k(X)$ is the number of words of length k that appear in X. The entropy dimension of a (one-dimensional) subshift with positive entropy is of course 1.

Lemma 3.11

Let X be a subshift with lower entropy dimension less than $1/d$. If $f \in \mathrm{Aut}(X)$ satisfies $|f^n| = O(\log^d n)$, then f is periodic.

Proof. Suppose we have $|f^n| = O(\log^d n)$ for large n. Then the radius of f^n is also $O(\log^d n)$. It follows that the trace subshift of f has complexity function at most $n \mapsto N_{\lfloor C\log^d n\rfloor}(X)$ for some constant C. If f is not of finite order, by the Morse-Hedlund theorem we must have $N_{\lfloor C\log^d n\rfloor}(X) > n$ for all n. Substituting $\lfloor e^{\sqrt[d]{n/C}}\rfloor$ for n we get $N_n(X) \geq e^{\sqrt[d]{an}}$ for some constant $a > 1$. Substituting this lower bound into the definition of lower entropy dimension, we get $\underline{D}(X) \geq \frac{1}{d}$. □

Theorem 3.12

The group $\mathrm{Aut}(A^{\mathbb{Z}})$ has a finitely-generated subgroup G such that every subshift X with $G \leq \mathrm{Aut}(X)$ has lower entropy dimension at least $1/4$.

Theorem 3.12 is of course an immediate corollary of Lemma 3.11. It states a "low-complexity restriction" on the automorphism group, i.e. it states that automorphism groups of subshifts with low enough complexity (growth of the number of admissible words) cannot have some property. The above theorem seems to be the first "low-complexity restriction" on automorphism groups where

1. the complexity bound is superpolynomial,
2. there are no additional dynamical restrictions, and
3. the prevented behavior can be exhibited in the automorphism group of another subshift.

There are previously known restrictions satisfying any two of these items. For 1.&2., zero entropy prevents exponential distortion [CFKP18]; for 1.&3., [CK16] shows that if X is minimal and has upper entropy dimension less than $1/2$, then it is amenable (while $\mathrm{Aut}(A^{\mathbb{Z}})$ is not); for 2.&3. (very low complexity restrictions) there are many results, see [PS22].

3.5.2 Distortion in Turing machine groups

We recall the definition of the group of Turing machines from [BKS16].

Definition 3.13

Let $n \geq 2$ and $k \geq 1$. Let Y_n be the full shift on n letters, and $X_k = \{x \in \{0,1,...,k\}^{\mathbb{Z}} \mid 0 \notin \{x_i, x_j\} \implies i = j\}$. Then

$$\mathrm{RTM}(n,k) = \{f \in \mathrm{Aut}(Y_n \times X_k) \mid f|_{Y_n \times \{0^{\mathbb{Z}}\}} = \mathrm{id}|_{Y_n \times \{0^{\mathbb{Z}}\}}\}.$$

Theorem 3.14

Let $n \geq 2, k \geq 1$. Then the group of Turing machines $\mathrm{RTM}(n,k)$ contains a distortion element; indeed there is a finitely-generated subgroup $G = \langle F \rangle$ and an element f such that $|f^n|_F = \log^4 n$.

Proof. We show that it immediately follows from the main theorem that $\mathrm{RTM}(72, 384)$ has a distortion element. We then explain how to conclude this for all $\mathrm{RTM}(n,k)$.

Recall that our automorphisms use the alphabet

$$\Sigma_\star = \left(\Gamma_\star^2 \times \{+1,-1\}\right) \sqcup \left((Q_\star \times \Gamma_\star) \times \Gamma_\star\right) \sqcup \left(\Gamma_\star \times (Q_\star \times \Gamma_\star)\right)$$

where $\Gamma_\star = \Gamma \times \{0,1\}$ and $Q_\star = Q \times \{d_\rightarrow, d_\leftarrow\} \times \left(\{+1,-1\} \times \{0,1\}^2 \times \Gamma\right)$.

We may instead view this as

$$(\Gamma_\star^2 \times \{+1,-1\}) \sqcup (Q_\star \times \{\uparrow,\downarrow\} \times \Gamma_\star^2),$$

by grouping $(Q_\star \times \Gamma_\star^2)$ and $(\Gamma_\star \times (Q_\star \times \Gamma_\star))$ together and replacing the choice with an arrow from $\{\uparrow,\downarrow\}$. Next, moving $\{\uparrow,\downarrow\}$ to the state and dropping $\{+1,-1\}$ out of it, we may view this as

$$(\Gamma_\star^2 \times \{+1,-1\}) \sqcup (Q' \times \{+1,-1\} \times \Gamma_\star^2),$$

for a certain set of states Q' with $|Q'| = |Q_\star| = 384$.

Consider the sofic subshift Z where a symbol of $(Q' \times \{+1,-1\} \times \Gamma^2)$ can appear at most once. We clearly have a conjugacy $Z \cong X_{384} \times Y_{72}$, since $|\Gamma_\star^2 \times \{+1,-1\}| = 72$.

It is easy to see that all of the generators F defined in Lemma 3.9 fix Z. Furthermore, our generators only act near the head, so by definition this restricted action makes them elements of $\mathrm{RTM}(72,384)$. The element $f_{\mathcal{M}_s}$ coming from the SMART machine clearly has infinite order, since it acts as the SMART machine on infinite configurations. The word norm of $f_{\mathcal{M}_s}$ w.r.t. F of course cannot grow faster after restricting these elements to an invariant subspace, so we obtain that the subgroup of $\mathrm{RTM}(72,384)$ generated by F still has a distortion element, and the distortion is at least as bad as on the full shift.

Now, we describe some minor modifications to the main construction that allow to conclude the result for $\mathrm{RTM}(n,k)$. In the construction of the main theorem, in place of the alphabet recalled above, take any finite set S and use instead

$$((\Gamma^2 \times \{+1,-1\}) \sqcup S) \sqcup (Q' \times ((\Gamma^2 \times \{+1,1\}) \sqcup S)).$$

Imagining elements of S as new empty conveyor belts of size 1, it is clear how most generators of F should act, as their action is defined by how they act on finite conveyor belts. The element θ does not respect the conveyor belts, but it is also clear how it should act (now that we have moved $\{+1,-1\}$ out of the state onto the tape) – it simply moves all heads.

Now recall that the only use of θ was to make sure that the automorphisms $f_{\mathrm{cb},\to t}$ (resp. $f_{\mathrm{cb},t\leftarrow}$ and $f_{\mathrm{cb},t\leftrightarrow t}$) are in our group. These automorphisms apply the involution $(+1) \leftrightarrow (-1)$ on the sign carried either by $\Gamma_*^2 \times \{+1,-1\}$ or by a head, at distance t to the right of the heads (resp. left of the heads, resp. both left and right of the heads). The correct extension of these is simply that the flip $(+1) \leftrightarrow (-1)$ does nothing on symbols in S. Then θ allows the implementation of natural analogs of the automorphisms $f_{\mathrm{cb},\to t}$, $f_{\mathrm{cb},t\leftarrow}$ and $f_{\mathrm{cb},t\leftrightarrow t}$ (with the exact same description).

Next, we recall that the automorphisms $f_{\mathrm{cb},\to t}$ are only used "through conjugation", i.e. they are used during the two-scale trick in very specific situations where we already know the head is on a large conveyor belt, in particular there are no S-symbols in the affected part. Thus the proof goes through without any modifications.

The introduction of S with $|S| = t$ changes the group of Turing machines from $\mathrm{RTM}(72,284)$ to $\mathrm{RTM}(72+t,384)$, and $\mathrm{RTM}(72+t,384)$ clearly embeds in $\mathrm{RTM}(72+t,384+\ell)$ for any $\ell \geq 0$ (by behaving as the identity when the head is in one of the ℓ many new states). In particular for large enough m, we can pick t,ℓ such that $72+t = n^m$ and $384+\ell = kn^m$, to get a distortion element in a subgroup of $\mathrm{RTM}(n^m, kn^m)$. Finally, there is an embedding of $\mathrm{RTM}(n^m, kn^m)$ into $\mathrm{RTM}(n,k)$, by considering m-blocks of cells as individual cells, and considering the word on the tape at the origin as part of the state (indeed this is an isomorphism). □

3.5.3 Distortion in higher-dimensional Brin-Thompson mV

It was shown by Belk and Bleak that classical reversible Turing machines embed in the Brin-Thompson group $2V$. More generally, the group of Turing machines embeds in $2V$, and indeed this embedding is entirely transparent. For this, we recall the *moving-tape model* of Turing machines.

Definition 3.15

Write $\mathrm{RTM}_{\mathrm{fix}}(n,k)$ for the family of homeomorphisms $f : [\![k]\!] \times [\![n]\!]^{\mathbb{Z}} \to [\![k]\!] \times [\![n]\!]^{\mathbb{Z}}$ such that for some radius $r \geq 1$ and local rule $f_{\mathrm{loc}} : \{0,1\}^r \times \{0,1\}^r \times [\![k]\!] \to \{0,1\}^* \times \{0,1\}^* \times [\![k]\!]$ we have

$$f(xu.vy, a) = (xu'.v'y, b) \text{ whenever } f_{\mathrm{loc}}(u,v,a) = (u',v',b)$$

and for all u, v, $f_{\mathrm{loc}}(u,v) = (u',v',n)$ satisfies $|u'| + |v'| = 2r$.

A proof of the following easy fact was outlined in [BKS16]; one simply translates tape shifts into head movement into the opposite direction.

Lemma 3.16

The family of homeomorphisms $\mathrm{RTM}_{\mathrm{fix}}(n,k)$ forms a group under composition, and there is a canonical group isomorphism $\mathrm{RTM}_{\mathrm{fix}}(n,k) \cong \mathrm{RTM}(n,k)$.

Lemma 3.17

The group $\mathrm{RTM}(n,k)$ embeds in the Brin-Thompson group mV for all $m \geq 2, n \geq 2, k \geq 1$.

As I did not contribute to this result, which is very similar to the proof in [BB17], and was also essentially stated in [BKS16], I only give an outline of the proof written by Ville Salo:

Proof. The group $2V$ embeds in mV, so it is enough to show this for $m = 2$. First, it is enough to embed $\mathrm{RTM}(n,1)$, since $\mathrm{RTM}(n,k)$ embeds in $\mathrm{RTM}(n,k+\ell)$ for all $\ell \geq 0$, thus in particular in $\mathrm{RTM}(n,n^m)$ for sufficiently large m, and this group is isomorphic to $\mathrm{RTM}(n,1)$ (see the end of the proof of Theorem 3.14).

Now pick a complete suffix code $C \subset \{0,1\}^*$ of cardinality n, and a complete prefix code $D \in \{0,1\}^*$ of cardinality n. One can uniquely parse any $x.y \in \{0,1\}^{\mathbb{Z}}$ as $\cdots u_{-2}u_{-1}.v_0v_1v_2\cdots$ with $u_{-i} \in C, v_i \in D$ for all applicable i, which gives a homeomorphism $\phi : \{0,1\}^{\mathbb{Z}} \to [\![n]\!]^{\mathbb{Z}}$. For $g \in \mathrm{RTM}(n,1)$, the map g^ϕ is easily seen to be in $2V$, so this gives a group-theoretic embedding of $\mathrm{RTM}(n,1)$ into $2V$. □

Dynamically, the proof gives a topological conjugacy between the natural action of $\mathrm{RTM}(n,1)$ and the natural action of the subgroup of $2V$ that respects the encoding.

Then, Theorem 3.4 and the aforementioned embedding lead to the following corollary:

Theorem 3.18

The Brin-Thompson group mV contains a distortion element.

Proof. The embedding of the group $\mathrm{RTM}(n,k)$ in particular embeds the group where we constructed a distortion element. Adding the finite generating set of mV clearly cannot make the element less distorted. □

This theorem provides a new restriction for geometries of $2V$. Namely, it is known that Thompson's group V admits a proper action by isometries on a $\mathrm{CAT}(0)$ cube complex [Far05]. By [Hag21, Theorem 1.5], a group with distortion elements does not admit such an action, thus:

Corollary 3.19

The Brin-Thompson group mV does not act properly on a $\mathrm{CAT}(0)$ cube complex for $m \geq 2$.

We thank Anthony Genevois for pointing out this corollary.

3.6 Open questions

Question 1. *Are there ever distortion elements in* $\mathrm{Aut}(X)$ *for* X *a minimal subshift? What about* X *of zero entropy?*

Minimal subshifts are interesting, because at present we do not know any restrictions on their automorphism groups, yet all known examples are locally virtually abelian. Zero entropy is interesting because on the one hand there are many known examples of interesting behaviors in their automorphism groups, but [CFKP18] shows that exponential distortion is impossible.

Next, it seems worth recalling the remaining parts of [CFKP18, Questions 5.1–3].

Question 2. *Are there more natural subgroups having distortion elements in* $\mathrm{Aut}(A^{\mathbb{Z}})$*, or even in* $\mathrm{Aut}(X)$*, where* $X \subset A^G$ *is an arbitrary subshift on an abelian group* G*? For example, can we embed the Heisenberg group (or more generally* $\mathrm{SL}(3,\mathbb{Z})$*), or the Baumslag-Solitar group* $\mathrm{BS}(1,n)$*?*

The Heisenberg group was originally asked about in [KR90] (though not explicitly due to distortion concerns). One important note about this group is that every (infinite f.g. torsion-free nonabelian) nilpotent group contains a copy of it. Nilpotent groups are considered some of the simplest (in the non-technical sense) kinds of infinite groups after abelian groups; in the case of automorphism groups of subshifts we can implement a wide variety of behaviors, yet embeddability of nilpotent groups remains a mystery.

Embedding the Baumslag-Solitar group is the same as finding an element of infinite order that is conjugate to a higher power of itself. We believe the SMART machine does not have this property (before or after an embedding into $\mathrm{Aut}(A^{\mathbb{Z}})$), but we have not proved this.

A slightly more abstract question of interest is whether there exists an amenable subgroup of the automorphism group of a subshift which has distortion elements. One thing amenability rules out is groups that are too large, e.g. f.g.-universal subgroups. The Heisenberg group and $\mathrm{BS}(1,n)$ are of course amenable (even solvable).

Question 3. *Can a one-sided subshift have distortion elements in its automorphism group? Does* $\mathrm{Aut}(A^{\mathbb{N}})$ *have distortion elements?*

Note that $\mathrm{Aut}(A^{\mathbb{N}})$ is simply the subgroup of $\mathrm{Aut}(A^{\mathbb{Z}})$ consisting of automorphisms f such that both f and f^{-1} have "one-sided radius", i.e. $f(x)_i$, $f^{-1}(x)_i$ depend only on $x_{[\![i,i+r]\!]}$ for some r. We do not even know whether one-sided automorphisms of subshifts can have sublinear radius growth.

Question 4. *What are the distortion functions (word norm growth rates) of elements of* $\mathrm{Aut}(A^{\mathbb{Z}})$ *(or* $\mathrm{Aut}(X)$ *for more general subshifts)?*

Of course, in the case of a non-finitely generated group like $\mathrm{Aut}(A^{\mathbb{Z}})$, the distortion function depends on the finite generating set chosen. While it is of interest to implement distortion functions with respect to subgroups, a more natural object to consider is the directed set of distortion functions with increasing generating sets.

Similar questions can be asked about groups of Turing machines and the Brin-Thompson $2V$, where we also exhibit elements whose word norm grows polylogarithmically, but have no further control on the distortion.

A natural idea for getting control over the distortion function would be to use, in place of SMART, a general-purpose Turing machine, which is made distorted using the reversible Hooper trick from [KO08] (and finally embedded in some natural way into the automorphism group of a subshift). It is known that this construction always produces Turing machines with zero Lyapunov exponents, i.e. with sublinear movement [GS17; Jea14].

Question 5. *Does the Kari-Ollinger construction in [KO08] always produce distortion elements?*

4

Surface entropies, aperiodicity and group confluence

As mentioned in the general introduction, another direction of study in symbolic dynamics is the generalization of subshifts *on groups*: for G a group, G *subshifts* (or subshifts for short) are the closed and shift-invariant subsets of Σ^G, for a finite alphabet Σ.

The geometry of the group G then has a strong influence on the properties of subshifts having G as a surface. For example, in terms of (a)periodicity, a lot of properties are not known. On SFTs[1] on $\mathbb{Z}$ or $\mathbb{Z}^2$, weak and strong aperiodicity are equivalent: the former asks configurations to have infinitely many distinct translates, while the latter forbids any configurations to have a single non-trivial period vector. This equivalence breaks over $\mathbb{Z}^3$, but for many groups this potential equivalence is unknown.

Another direction of study is a classification of the groups having aperiodic SFTs, i.e. SFTs that only contain aperiodic configurations. While for some groups such examples of aperiodic SFTs are known or proved to be impossible, for many groups, their existence is still open.

In this chapter, we consider a different direction: instead of asking for all configurations to be aperiodic, can we ask for a sufficient condition for at least one aperiodic configuration to exist? This was the point of view I had already developed in [CH22, Proposition 25] with Benjamin Hellouin de Menibus, which proved that a subclass of subshifts on $\mathbb{Z}^d$ with high complexity have to contain at least one strongly aperiodic configuration. However, despite our best efforts during the 2021 winter break, we could not generalize nor improve this partial result.

Ville Salo came to our rescue, and generalized [CH22, Proposition 25] to any $\mathbb{Z}^d$ subshift. With Benjamin Hellouin, we later improved this generalization to m-entropies and $(d-m)$-aperiodicity. The most interesting part, though, was another generalization conducted with Ville: could we obtain a similarly minded result on subshifts on groups? For which classes of groups?

Section 4.1 develops the generalization of [CH22, Proposition 25] obtained with Benjamin Hellouin and Ville Salo in the case of $\mathbb{Z}^d$ subshifts. Instead of proving the full statement, we prove one particular case, which highlights the key idea of Ville Salo.

In Section 4.2, we create a notion we call *group confluence*, which generalizes the key ingredient of the proof of Section 4.1 on $\mathbb{Z}^d$ to orderable groups. We then prove in Corollary 4.9 that some class of groups (namely, the poly–(torsion-free abelian groups)) are confluent.

[1] Subshift of finite type, i.e. defined by finitely many forbidden patterns.

4.1 Entropies and aperiodicity in $\mathbb{Z}^d$-subshifts

4.1.1 Generalization on $\mathbb{Z}^d$ subshifts

In what follows, a $\mathbb{Z}^d$ subshift X is a closed and shift-invariant subset of $\Sigma^{\mathbb{Z}^d}$ for a finite alphabet Σ. Equivalently, there exists a family of forbidden patterns $\mathcal{F}$ such that X is the set of configurations $x \in \Sigma^{\mathbb{Z}^d}$ that do not contain any pattern of $\mathcal{F}$.

Definition 4.1 Periodicity of $\mathbb{Z}^d$-configurations

1. A configuration $x \in \Sigma^{\mathbb{Z}^d}$ is $(v_1, \dots, v_i)$-periodic, for $v_1, \dots, v_i \in \mathbb{Z}^d$, if for any $0 \leq j \leq i, \sigma^{v_j}(x) = x$.
2. A configuration $x \in \Sigma^{\mathbb{Z}^d}$ is i-periodic if there exists a family of linearly independent vectors $(v_1, \dots, v_i) \in (\mathbb{Z}^d)^i$ such that x is $(v_1, \dots, v_i)$-periodic.
3. A configuration $x \in \Sigma^{\mathbb{Z}^d}$ is (strongly) *aperiodic* if it is not v-periodic for any $v \in \mathbb{Z}^d$.

Definition 4.2 Complexity

For X a $\mathbb{Z}^d$ subshift, we define the complexity function $N_X(n)$ as the number of globally admissible patterns of domain $[\![0, n]\!]^d$ that appear in X.

Definition 4.3

Let X be a $\mathbb{Z}^d$ subshift. For $m \leq d$, define the m-entropy of X as:

$$h_m(X) = \limsup_{n \mapsto +\infty} \frac{\log N_X(X)}{n^m} \in \mathbb{R} \cup \{+\infty\}$$

For $m = d$, $h_d(X)$ is called the topological entropy of X. For $m = d - 1$, $h_{d-1}(X)$ is called the surface entropy of X.

We Benjamin Hellouin de Menibus and Ville Salo, we proved the following theorem:

Theorem 4.4

Let X be any $\mathbb{Z}^d$ subshift. If $h_m(X) = +\infty$, then there exists a configuration $x \in X$ that is $(d - m)$-aperiodic (i.e. for any family of $d - m$ linearly independent vectors $(v_1, \dots, v_{d-m}) \in (\mathbb{Z}^d)^{d-m}$, x is not $(v_1, \dots, v_{d-m})$-periodic).

This has an immediate corollary:

Corollary 4.5

Let X by any $\mathbb{Z}^d$ subshift. If $h_{d-1}(X) = +\infty$, then X contains a strongly aperiodic configuration.

In what follows, we only prove Corollary 4.5. Indeed, it was the first statement we had, and it is the statement that we generalize in Section 4.2 from $\mathbb{Z}^d$ subshifts to G subshifts, for some class of groups G.

The key idea of this proof is due to Ville Salo.

4.1.2 Proof of Corollary 4.5

Define:

$$\begin{aligned} B_d(n,k) =& [\![0, n+2k]\!]^d \\ F_d(n,k) =& \left\{(i_1,\dots,i_d) \in [\![0,n+2k]\!]^d : \forall j \in [\![1,d]\!], i_j < k \text{ or } i_j > n+k\right\} \\ I_d(n,k) =& \left\{(i_1,\dots,i_d) \in [\![0,n+2k]\!]^d : \forall j \in [\![1,d]\!], k \leq i_j \leq n+k\right\} \end{aligned}$$

Of course, $B_d(n,k)$ is the disjoint union of $F_d(n,k)$ and $I_d(n,k)$. We now prove that for a subshift X of infinite surface entropy, there exist patterns of arbitrary size which agree on increasingly thick borders:

Lemma 4.6

If $h_{d-1}(X) = +\infty$, then for any $k \in \mathbb{N}$, there exists two globally admissible patterns $w(k) \neq w'(k)$ of X, both with domain $B_d(n,k)$, such that for any $i \in F_d(n,k)$, $w(k)_i = w'(k)_i$.

Proof. Let $k \in \mathbb{N}$ be fixed, and recall that a d-dimensional hypercube has $2d$ facets of dimension $d-1$. Then the number $N_{X,F}(n,k)$ of patterns of domain $F_d(n,k)$ is at most

$$N_{X,F}(n,k) \leq |\Sigma|^{2d\cdot k(n+2k)^{d-1}}$$

On the other hand, the number of admissible cubes of domain $[\![0,n+2k]\!]^d$ is $N_X(n+2k)$. As $\log N_X(n+2k)/(n+2k)^{d-1}$ is unbounded when k is fixed and n goes to infinity, there must exist some $n \in \mathbb{N}$ such that $\log(|\Sigma|)\cdot 2d\cdot k < (\log N_X(n+2k))/(n+2k)^{d-1}$, i.e.

$$N_{X,F}(n,k) \leq |\Sigma|^{2dk(n+2k)^{d-1}} < N_X(n+2k)$$

We conclude by the pigeonhole principle. □

Proof of Corollary 4.5

Proof. Let $w(k) \neq w'(k)$ be a family given by the previous lemma. Now, for every $k \in \mathbb{N}$, define $i^{(k)} \in [\![0,n+2k]\!]^d$ as the maximal (for the lexicographic order) position $i = (i_1,\dots,i_d)$ such that $w(k)_i \neq w'(k)_i$. Then, we have the two following properties:

$$\begin{aligned} &\forall k \in \mathbb{N},\ w(k)_{i^{(k)}} \neq w'(k)_{i^{(k)}} \\ &\forall k \in \mathbb{N}, \forall j \in [\![0,n+2k]\!]^d, j >_{lex} i^{(k)} \implies w(k)_j = w'(k)_j \end{aligned}$$

By compactness on the sequences of globally admissible patterns $\sigma^{i^{(k)}}(w(k))$ and $\sigma^{i^{(k)}}(w'(k))$, there exists two configurations $x, x' \in X$ such that:

$$x_0 \neq x'_0 \tag{4.1}$$

$$\forall j \in \mathbb{Z}^d, j >_{lex} 0 \implies x_j = x'_j \tag{4.2}$$

By contradiction, assume there exists two linearly independent vectors $u, v \in \mathbb{Z}^d$ such that x is u-periodic, and x' is v-periodic. As for $p \in \mathbb{Z}^d \setminus \{0\}$, either $p >_{lex} 0$ or $-p >_{lex} 0$, we assume $u >_{lex} 0$ and $v >_{lex} 0$. Then, using u and v-periodicities and property (4.2), we obtain:

$$x_0 = x_u = x'_u = x'_{u+v} = x_{u+v} = x_v = x'_v = x'_0$$

which contradicts property (4.1). □

4.2 Group confluence

In the proof of Theorem 4.4, we use the fact that two linearly independent vectors $u, v \in \mathbb{Z}^d$ (with $u, v >_{\text{lex}} 0$) generate a semi-infinite positive lattice that joins u and v in the strictly positive domain[2]. We call this property *confluence*, and this section proves that some class of groups are confluent: namely the poly–(torsion free abelian) groups.

4.2.1 Main result

Formally, let $(G, \cdot)$ be a group equipped with a left-invariant total order $\leq$. (This means that for $a, b, c \in G$, if $a \leq b$, then $c \cdot a \leq c \cdot b$). We denote $G_+ = \{g \in G \mid g > 1_G\}$ and $G_- = \{g \in G \mid g < 1_G\}$.

For any word $w \in G^*$, we denote $[w] = w_0 \cdot w_1 \cdots w_{\text{len}(w)-1}$ the corresponding element of G. The operation $w \in G^* \mapsto [w] \in G$ is a morphism: for $w_1, w_2 \in G^*$, $[w_1 \cdot {w_2}^{-1}] = [w] \cdot [w_2]^{-1}$ where, for any word $w \in G^*$, w^{-1} denotes $w_{\text{len}(w)-1}^{-1} \cdots w_1^{-1} \cdot w_0^{-1}$.

We say that $w \in G^*$ is a *positive sequence* if for every non-empty strict prefix p of w, $[p] > 1_G$. And we say that $w \in G^*$ is a *positive path* if it is a positive sequence satisfying $[w] = 1_G$. Note that if w is a positive path, then w^{-1} is also a positive path.

Definition 4.7

An ordered group $(G, \leq)$ is *confluent* if for any $a, b \in G_+$, there exists some $u \in \{a, b, a^{-1}, b^{-1}\}^*$ such that $a \cdot u \cdot b^{-1}$ is a positive path.

Of course, if $a = b$, then this definition is trivial. More interestingly, if G is torsion free abelian (every such abelian group can be equipped with a left-order), then $aba^{-1}b^{-1}$ is a positive path.

The main result we prove in this section is that confluence is a property stable by group extensions[3]:

Theorem 4.8

Let $1 \mapsto K \mapsto G \mapsto H \mapsto 1$ be an exact short sequence such that $(G, \leq)$ is a left-ordered group which correctly defines a left-order $\preceq$ on H (i.e. if $\pi : G \mapsto H$ is the projection from G to H, then $1_G \leq x \implies 1_H \preceq \pi(x)$).
If both K and H are confluent, then G is confluent.

Groups obtained by iterating group extensions from a class C are called *poly-C*. As torsion free abelian groups are left-orderable and confluent by a previous remark, we obtain by induction on theorem 4.8:

Corollary 4.9

Let $(G, \leq)$ be a poly–(torsion free abelian) group. Then G is left-orderable with a confluent order.

Examples of such groups are torsion free nilpotent groups, strongly polycyclic groups, or wreath-products of torsion free abelian groups.

[2] As $\mathbb{Z}^d$ is abelian, the previous sentence translates into $u, v, u+v, v+u > 0$ and $u+v = v+u$. Of course, we are interested in generalizing this to non–abelian groups as well.

[3] Obviously, readers unfamiliar with the technicalities of group extensions will skip the mathematical details and proofs of the following paragraphs. Let us just say the following: group extensions are one key method used to build classes of groups. For example, nilpotent groups are obtained by iterating central extensions from abelian groups; polycyclic groups are obtained by iterating groups extensions from cyclic groups.

4.2.2 Proof of Theorem 4.8

With the same notations, let $a, b \in G_+$.

If $a, b \in K$

By confluence in K, there exists $u \in \{a, b, a^{-1}, b^{-1}\}^*$ such that $a \cdot u \cdot b^{-1}$ defines a positive path. Then $a \cdot u \cdot b^{-1}$ is also a positive path in G, and we obtain confluence for a and b.

If $a, b \notin K$

First step: confluence in H

First, we have $\pi(a), \pi(b) \in H_+$ (because $\preceq$ is induced by $(G, \leq)$ on H). By confluence of H, there exists $\tilde{u} \in \{\pi(a), \pi(b), \pi(a)^{-1}, \pi(b)^{-1}\}^*$ some positive path in H such that $\pi(a) \cdot \tilde{u} \cdot \pi(b)^{-1} = 1_H$. Considering $u \in \{a, b, a^{-1}, b^{-1}\}^*$ the sequence corresponding to $\tilde{u} \in \{\pi(a), \pi(b), \pi(a)^{-1}, \pi(b)^{-1}\}^*$ in G, then $a \cdot u \cdot b^{-1}$ is a positive sequence, and $[a \cdot u \cdot b^{-1}] \in K$.

If $[a \cdot u \cdot b^{-1}] = 1_G$, then $a \cdot u \cdot b^{-1}$ is a positive path that proves confluence for a and b. In what follows, we assume $[a \cdot u \cdot b^{-1}] \neq 1_G$.

Define $A = a \cdot u \cdot b^{-1}$. Then either $[A] \in K_+$, or $[A^{-1}] \in K_+$. As both A and A^{-1} are positive sequences (because $\pi(a) \cdot \tilde{u} \cdot \pi(b)$ is a positive path in H, and the inverse of a positive path is also a positive path), by permuting a and b, we can assume that $A = a \cdot u \cdot b^{-1}$ is a positive sequence satisfying $[A] \in K_+$.
Now, define $B = b \cdot A \cdot b^{-1}$ if $[b \cdot A \cdot b^{-1}] \in K_+$, and $B = (b \cdot A \cdot b^{-1})^{-1}$ if $[b \cdot A \cdot b^{-1}] \in K_-$ (the case $B = 1_G$ is not possible, because $A \neq 1_G$). We prove that B defines a positive sequence such that $[B] \in K_+$. Indeed, by definition, $[B] \in K_+$, and:

- If $B = b \cdot A \cdot b^{-1}$, then consider any prefix p (non necessarily strict) of A: as $[p] \geq 1_G$ and $b > 1_G$, by left-invariance we obtain $[b \cdot p] \geq b > 1_G$. So any non-empty strict prefix of B defines an element of G_+, and B is a positive sequence.
- If $B = b \cdot A^{-1} \cdot b^{-1}$, then consider any *strict* prefix p of A^{-1}: as $[p] \geq 1_G$ (because A^{-1} is a positive sequence) and $b > 1_G$, we obtain $[b \cdot p] \geq b > 1_G$. The only remaining prefix of B we have to prove positive is $p = b \cdot A^{-1}$.
  As $[B] > 1_G$ and $b \geq 1_G$, we obtain by left-invariance $[B \cdot b] \geq [B] > 1_G$. And $[B \cdot b] = [b \cdot A^{-1} \cdot b^{-1} \cdot b] = [b \cdot A^{-1}] = [p]$, which proves that every non-empty strict prefix of B defines an element of G_+, i.e. that B is a positive sequence.

To conclude, we now have two positive sequences A and B that respectively start with a and b, such that A^{-1} and B^{-1} are also positive sequences, and $[A] \in K_+$ and $[B] \in K_+$.

Second step: confluence in K

By confluence in K, there exists $\tilde{U} \in \{[A], [B], [A]^{-1}, [B]^{-1}\}^*$ such that $[A] \cdot \tilde{U} \cdot [B]^{-1}$ is a positive path from $[A]$ to $[B]$ in K. We now prove that the de-substituted path $A \cdot U \cdot B^{-1}$ is also a positive path in G, for $U \in \{a, b, a^{-1}, b^{-1}\}^*$ the de-substitution of $\tilde{U}$ in which each letter $[X] \in \{[A], [B], [A]^{-1}, [B]^{-1}\}$ is replaced by its path $X \in \{A, B, A^{-1}, B^{-1}\}$.
Consider a non-empty strict prefix of $A \cdot U \cdot B^{-1}$. It decomposes into $P \cdot p$, for $P \in \{a, b, a^{-1}, b^{-1}\}^*$ the de-substitution of some strict prefix $\tilde{P} \in \{[A], [B], [A]^{-1}, [B]^{-1}\}^*$ of $\tilde{U}$, and p some strict prefix of A, B, A^{-1} or B^{-1}. If P is empty, then p is not, and $[p] > 1_G$ (because A, B, A^{-1} and B^{-1} are positive sequences), so $[P \cdot p] = [p] > 1_G$. If P is non-empty, then $[p] \geq 1_G$ (because it is a strict prefix of one of

the positive sequences A, B, A^{-1}, B^{-1}), and by left-invariance $[P \cdot p] \geq [P] > 1_G$. In any case, any prefix of $A \cdot U \cdot B^{-1}$ belongs in G_+.

As $[A \cdot U \cdot B^{-1}] = [A \cdot \tilde{U} \cdot B^{-1}] = 1_G$, the sequence $A \cdot U \cdot B^{-1} \in \{a, b, a^{-1}, b^{-1}\}^*$ is a positive path in G. As A and B respectively start with letters a and b, $A \cdot U \cdot B^{-1}$ starts with letter a and ends with letter b^{-1}. This proves confluence for a and b.

If $a \in K, b \notin K$ or $a \notin K, b \in K$

By symmetry, assume that $a \in K$ and $b \notin K$. Notice that $[a \cdot b \cdot a^{-1}] \neq 1_G$ (otherwise $b = 1_G \notin G_+$).

Define $a' = a \cdot b \cdot a^{-1}$ if $[a \cdot b \cdot a^{-1}] \in G_+$, or $a' = a \cdot b^{-1} \cdot a^{-1}$ if $[a \cdot b \cdot a^{-1}] \in G_-$. By the previous case, as $[a'] \notin K$, there exists some word $\tilde{u} \in \{[a'], b, [a']^{-1}, b^{-1}\}^*$ such that $[a'] \cdot \tilde{u} \cdot b^{-1}$ is a positive path in G: and we now prove that the de-substituted path $a' \cdot u \cdot b^{-1} \in \{a, b, a^{-1}, b^{-1}\}^*$ of $[a'] \cdot \tilde{u} \cdot b^{-1}$ is a positive path in G.

First, $[a' \cdot u \cdot b^{-1}] = [[a'] \cdot \tilde{u} \cdot b^{-1}] = 1_G$.

Now, consider a non-empty strict prefix of $a' \cdot u \cdot b^{-1}$ in $\{a, b, a^{-1}, b^{-1}\}^*$. If it is a prefix (non-strict) of a', then it is positive: indeed, $a > 1_G$ by definition, $a \cdot b \geq a > 1_G$ by left-invariance, $a' > 1_G$ by definition, and if $a' = a \cdot b^{-1} \cdot a^{-1}$, then $[a \cdot b^{-1}] = [a'] \cdot a \geq [a'] > 1_G$ by left-invariance.

Now, consider a strict prefix of length greater than 3. It decomposes into $P \cdot p$, where $P \in \{a, b, a^{-1}, b^{-1}\}^*$ is the de-substitution of some prefix $\tilde{P} \in \{[a'], b, [a']^{-1}, b^{-1}\}^*$ of $[a'] \cdot \tilde{u}$, and p is a strict prefix of either a', b, a'^{-1} or b^{-1}.

- If p is empty, then by hypothesis $[P \cdot p] = [P] > 1_G$ because $[a'] \cdot \tilde{u} \cdot b^{-1}$ is a positive path.

In particular, assume p is non-empty: it must be a strict prefix of either $a \cdot b \cdot a^{-1}$ or $a \cdot b^{-1} \cdot a^{-1}$:

- Assume p has length 1. Then $p = a$. As $[a'] \cdot \tilde{u} \cdot b^{-1}$ is a positive path, we have $[P] > 1_G$, and by left-invariance we obtain $[P \cdot p] = [P \cdot a] \geq [P] > 1_G$.
- Assume p has length 2. Define $\tilde{P'} = \tilde{P} \cdot (p \cdot a^{-1}) \in \{[a'], b, [a']^{-1}, b^{-1}\}^*$, which is also a prefix of $[a'] \cdot \tilde{u} \cdot b^{-1}$. Then $\tilde{P'}$ is a strict prefix of $[a'] \cdot \tilde{u} \cdot b^{-1}$, as the latter ends by b^{-1} and not $[a']$ or $[a']^{-1}$. So $[\tilde{P'}] > 1_G$, and denoting $P' \in \{a, b, a^{-1}, b^{-1}\}^*$ the de-substitution of $\tilde{P'}$, this means $[P'] > 1_G$.

  Then, as $a \geq 1_G$, we obtain by left-invariance that $[P' \cdot a] \geq [P'] > 1_G$, while $[P' \cdot a] = [P \cdot (p \cdot a^{-1}) \cdot a] = [P \cdot p]$.

This proves that $a' \cdot u \cdot b^{-1} \in \{a, b, a^{-1}, b^{-1}\}^*$ is a positive path in G, which starts with a and ends with b^{-1}. This proves confluence for a and b in G.

4.2.3 Consequences

We now properly define subshifts using groups as surfaces. Let G be a group, a Σ a finite alphabet. The set Σ^G equipped with the product topology becomes a Cantor space. G naturally acts on the left of Σ^G by $(g \cdot x)_h = x_{g^{-1}h}$, and we call this G action the *shift*.

Definition 4.10 — G subshift

A G *subshift* (or subshift for short) is a closed and shift-invariant subset of Σ^G.

To obtain a generalization of Corollary 4.5, we need to generalize the definition of surface entropy. We suggest the following one:

Definition 4.11 Infinite surface entropy

A G subshift X has *infinite surface entropy* if:

$$\forall n > 0, \forall B \Subset G, \exists F \Subset G, \#X|_F > n^{\#\partial_B F}$$

where $\partial_B F = \{g \in F \mid Bg \not\subseteq F\}$ is the *inner boundary*.

Informally, a subshift X has infinite surface entropy if for every possible closed boundary B, there exists a closed subset F such that $\partial_B F$ (the inner boundary of F by B) does not determine the interior of F: there are more patterns in X with domain F than boundaries of F. Unfortunately, this definition of infinite surface entropy differs from Definition 4.3: the former on groups uses arbitrary shapes for F and B, while the latter on $\mathbb{Z}^d$ is restricted to hypercubes.

We plan to investigate this gap in the future. However, it should be noted that on $\mathbb{Z}^d$, there is no obvious way to generalize Theorem 4.4 with arbitrary shapes instead of hypercubes: k-dimensional facets of a hypercube (for $k < d$) are well-known and defined, and they coincide well with subgroups of $\mathbb{Z}^d$; but there is no obvious definition of k-dimensional facets of an arbitrary shape.

In any way, using Definition 4.11, the obvious generalization of Corollary 4.5 follows by a morally identical argument:

Theorem 4.12

Let G be a group admitting a confluent right-invariant order. Then every subshift X of infinite surface entropy contains a strongly aperiodic configuration (i.e. a configuration $x \in X$ such that $\forall g \neq 1_G, g \cdot x \neq x$).

4.2.4 Questions

Let us now discuss a few points. First, Definition 4.11 has obvious links with the Følner condition of amenability for groups. And in fact, we can easily prove that if G is a non-amenable group, then every subshift of G has finite surface entropy. However, amenability is not a sufficient condition for G subshifts with infinite surface entropy to have strongly aperiodic configurations, by an example of Ville Salo:

Example. *Let $G = \mathbb{Z} \times \mathbb{Z}_2$. Then G admits a subshift of infinite surface entropy with no strongly aperiodic points. Namely, take a $\mathbb{Z}$-full shift X, and define an action of $\mathbb{Z} \times \mathbb{Z}_2$ by $(n, a) \cdot x = \sigma^n(x)$. Clearly we have infinite surface entropy because boundaries of natural Følner sets are of constant size but the subshift has exponentially many patterns on them. Just as obviously, there are no strongly aperiodic points.*

However, $\mathbb{Z} \times \mathbb{Z}_2$ has torsion (i.e. elements of finite order). This leads to the following question, asking for a more interesting example:

Question 6. *Does there exist a torsion free amenable finitely-generated group G which admits a subshift of infinite surface entropy without any strongly aperiodic configurations?*

In the other direction, one key notion in the proof of Theorem 4.12 is the notion of *group confluence*, which looks like an interesting notion by itself. The obvious (probably very hard) question would be:

Question 7. *Which groups admit a confluent left-order?*

To relate this question to the previous paragraphs, such groups obviously cannot have torsion (because for any left-orderable group G, $0 < g \implies g < g^2$, so in particular $g^n \neq 1_G$). One direction to consider would be the following:

Question 8. *Does there exist a torsion free solvable group G that does not admit a confluent left-order?*

Recall that solvable groups are the poly–abelian groups, i.e. obtained by induction from abelian groups. Unfortunately, torsion free poly–abelian groups are not the poly–(torsion free abelian) groups: as a consequence, the proof methods developed for Theorem 4.8 do not work anymore, which leaves this question open.

Conclusion

Distortion in automorphism groups of subshifts

This ARPE originally aimed at solving the following question:

> Does there exist a subshift X such that $\mathrm{Aut}(X)$ contains distortion elements?
> What conditions on X ensure that $\mathrm{Aut}(X)$ contains distortion elements?

We answered this question by the affirmative: yes, every non-trivial full-shift does contain a distortion element (Theorem 3.4). We even obtained the following classification for sofic subshifts: a sofic subshift contains a distortion element if and only if it has positive entropy (Theorem 3.10).

While this solved an open question in symbolic dynamics since at least [KR90; CFKP18], this also led to interesting discussions with mathematicians from group geometry: indeed, Theorem 3.18 prevents the multidimensional Brin-Thompson groups to act properly on CAT(0) cube complexes (Corollary 3.19). While these consequences are above my understanding, it was interesting to see interactions between automorphism groups of subshifts and another branch of mathematics.

Open interesting follow-up questions are detailed in Section 3.6. Most of them are probably very difficult, but a few mathematicians seemed interested in working of them.

Surface entropies, aperiodicity and confluence

As a side project, we worked part-time on aperiodicity in subshifts on groups. Apart from [GHV18] and [CH22], which studied the problem on $\mathbb{Z}^d$, the existence of strongly aperiodic configurations (instead of requiring all configurations to be strongly aperiodic) had scarcely been studied, and up to my knowledge it had never been studied for subshifts on groups.

> Does infinite surface entropy implies the existence of
> (strongly) aperiodic configurations?

We had left this question open with Benjamin Hellouin de Menibus a year earlier while working on [CH22]. During this ARPE, we answered with Ville Salo by the affirmative for $\mathbb{Z}^d$ subshifts in Corollary 4.5; furthermore, we generalized the notion to other classes of groups than $\mathbb{Z}^d$, and we created the notion of *group confluence*. This notion naturally generalizes one of the key ingredients of the aforementioned proof, namely: if you consider two positive elements a and b of a group, can you form two paths aw and bw' that "join" a and b in the positive cone?

This notion does not relate much to existing literature, but we leave a few open questions in Section 4.2.4.

Bibliography

[AB09] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009. ISBN: 978-0521424264 (cit. on p. iii).
DOI: 10.1017/CBO9780511804090.
URL: https://books.google.fi/books?id=8Wjqvsoo48MC.

[Bar89] David A. Barrington. "Bounded-width polynomial-size branching programs recognize exactly those languages in NC1". In: *Journal of Computer and System Sciences* 38.1 (1989), pp. 150–164 (cit. on pp. 17, 20).
DOI: https://doi.org/10.1016/0022-0000(89)90037-8.

[BB17] James Belk and Collin Bleak. Some undecidability results for asynchronous transducers and the Brin-Thompson group $2V$. In: *Transactions of the American Mathematical Society* 369.5 (2017), pp. 3157–3172 (cit. on p. 46).
DOI: 10.1090/tran/6963.

[BCN02] Vincent D. Blondel, Julien Cassaigne, and Codrin Nichitiu. "On the presence of periodic configurations in Turing machines and in counter machines". In: *Theoretical Computer Science* 289.1 (2002), pp. 573–590 (cit. on p. 4).
DOI: 10.1016/S0304-3975(01)00382-6.

[BDM22] Nicolas Bitar, Sebastian Donoso, and Alejandro Maass. "Distortion in Automorphisms of Expansive Systems". In: *Automata and Complexity*. Ed. by Andrew Adamatzky. Springer International Publishing, 2022, pp. 21–41 (cit. on p. 30).
DOI: 10.1007/978-3-030-92551-2_3.

[BKS16] Sebastián Barbieri, Jarkko Kari, and Ville Salo. "The group of reversible Turing machines". In: *Cellular automata and discrete complex systems*. Vol. 9664. Lecture Notes in Computer Science. Springer, [Cham], 2016, pp. 49–62 (cit. on pp. 44–46).
DOI: 10.1007/978-3-319-39300-1_5.

[BLR88] Mike Boyle, Douglas Lind, and Daniel Rudolph. "The automorphism group of a shift of finite type". In: *Transactions of the American Mathematical Society* 306.1 (1988), pp. 71–114 (cit. on p. 29).
DOI: 10.1090/S0002-9947-1988-0927684-2.

[CC20] Serge Cantat and Yves de Cornulier. "Distortion in Cremona groups". In: *Annali della Scuola Normale Superiore di Pisa. Classe di Scienze. Serie V* 20.2 (2020), pp. 827–858 (cit. on pp. 1, 29).
DOI: 10.2422/2036-2145.201806_005.

[CF06] Danny Calegari and Michael H. Freedman. "Distortion in transformation groups". In: *Geometry and Topology* 10 (2006), pp. 267–293 (cit. on pp. 1, 29).
DOI: 10.2140/gt.2006.10.267.

[CFK19] Van Cyr, John Franks, and Bryna Kra. "The spacetime of a shift endomorphism". In: *Transactions of the American Mathematical Society* 371.1 (2019), pp. 461–488 (cit. on p. 30).
DOI: 10.1090/tran/7254.

[CFKP18] Van Cyr, John Franks, Bryna Kra, and Samuel Petite. "Distortion and the automorphism group of a shift". In: *Journal of Modern Dynamics* 13.1 (2018), pp. 147–161 (cit. on pp. 1, 29, 30, 32, 43, 44, 47, 56).
DOI: 10.3934/jmd.2018015.

[CH22] Antonin Callard and Benjamin Hellouin de Menibus. "The Aperiodic Domino Problem in Higher Dimension". In: *39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*. Vol. 219. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 19:1–19:15 (cit. on pp. 2, 3, 48, 56).
DOI: 10.4230/LIPIcs.STACS.2022.19.

[CK16] Van Cyr and Bryna Kra. "The automorphism group of a minimal shift of stretched exponential growth". In: *Journal of Modern Dynamics* 10.02 (2016), p. 483 (cit. on pp. 29, 44).
DOI: 10.3934/jmd.2016.10.483.

[COT17] Julien Cassaigne, Nicolas Ollinger, and Rodrigo Torres-Avilés. "A small minimal aperiodic reversible Turing machine". In: *Journal of Computer and System Sciences* 84 (2017), pp. 288–301 (cit. on pp. 1, 4, 6, 19, 32).
DOI: 10.1016/j.jcss.2016.10.004.

[CS22] Antonin Callard and Ville Salo. *Distortion element in the automorphism group of a full shift*. 2022 (cit. on p. 3).
DOI: 10.48550/ARXIV.2208.00685.

[CT20] Pablo Concha-Vega and Rodrigo Torres-Avilés. *A Binary Complete and Aperiodic Turing machine*. working paper or preprint. 2020 (cit. on p. 19).
URL: https://hal.archives-ouvertes.fr/hal-02545145.

[DDMP16] Sebastián Donoso, Fabien Durand, Alejandro Maass, and Samuel Petite. "On automorphism groups of low complexity subshifts". In: *Ergodic Theory and Dynamical Systems* 36.1 (2016), pp. 64–95 (cit. on p. 30).
DOI: 10.1017/etds.2015.70.

[Far05] Daniel S. Farley. "Actions of picture groups on CAT(0) cubical complexes". In: *Geometriae Dedicata* 110 (2005), pp. 221–242 (cit. on p. 46).
DOI: 10.1007/s10711-004-1530-z.

[FH06] John Franks and Michael Handel. "Distortion elements in group actions on surfaces". In: *Duke Mathematical Journal* 131.3 (2006), pp. 441–468 (cit. on pp. 1, 29).
DOI: 10.1215/S0012-7094-06-13132-0.

[GHV18] Anael Grandjean, Benjamin Hellouin de Menibus, and Pascal Vanier. "Aperiodic Points in $\mathbb{Z}^2$-subshifts". In: *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. Vol. 107. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 128:1–128:13 (cit. on p. 56).
DOI: 10.4230/LIPIcs.ICALP.2018.128.

[GK11] Świątosław R. Gal and Jarek Kędra. "On distortion in groups of homeomorphisms". In: *Journal of Modern Dynamics* 5.3 (2011), pp. 609–622 (cit. on pp. 1, 29).
DOI: 10.3934/jmd.2011.5.609.

[GL19] Nancy Guelman and Isabelle Liousse. "Distortion in groups of affine interval exchange transformations". In: *Groups, Geometry, and Dynamics* 13.3 (2019), pp. 795–819 (cit. on pp. 1, 29).
DOI: 10.4171/GGD/505.

[Gro93] Mikhael Gromov. "Asymptotic invariants of infinite groups". In: *Geometric group theory, Vol. 2*. Ed. by Graham A. Niblo, Martin A. Roller, and J. W. S. Cassels. London Mathematical Society Lecture Note Series. Cambridge University Press, 1993 (cit. on pp. 1, 29).

[GS17] Pierre Guillon and Ville Salo. "Distortion in One-Head Machines and Cellular Automata". In: *Cellular Automata and Discrete Complex Systems, AUTOMATA 2017*. Vol. 10248. Lecture Notes in Computer Science. Springer, 2017, pp. 120–138 (cit. on pp. 4, 19, 30, 32, 47).
DOI: 10.1007/978-3-319-58631-1_10.

[Hag21] Frédéric Haglund. "Isometries of CAT (0) cube complexes are semi-simple". In: *Annales mathématiques du Québec* (2021), pp. 1–13 (cit. on p. 46).
DOI: 10.1007/s40316-021-00186-2.

[Hed69] Gustav A Hedlund. "Endomorphisms and automorphisms of the shift dynamical system". In: *Mathematical systems theory* 3.4 (1969), pp. 320–375 (cit. on p. 31).
DOI: 10.1007/BF01691062.

[Hoo66] Philip K. Hooper. "The Undecidability of the Turing Machine Immortality Problem". In: *The Journal of Symbolic Logic* 31.2 (1966), pp. 219–234 (cit. on p. 4).
DOI: 10.2307/2269811.

[Jea14] Emmanuel Jeandel. "Computability of the entropy of one-tape Turing machines". In: *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014)*. Vol. 25. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014, pp. 421–432 (cit. on p. 47).
DOI: 10.4230/LIPIcs.STACS.2014.421.

[KO08] Jarkko Kari and Nicolas Ollinger. "Periodicity and Immortality in Reversible Computing". In: *Mathematical Foundations of Computer Science 2008*. Springer Berlin Heidelberg, 2008, pp. 419–430 (cit. on pp. 4, 6, 19, 47).
DOI: 10.1007/978-3-540-85238-4_34.

[KR90] Ki Hang Kim and Fred William Roush. "On the automorphism groups of subshifts". In: *Pure Mathematics and Applications* 1.4 (1990), pp. 203–230 (cit. on pp. 1, 29, 32, 43, 47, 56).

[Kůr97] Petr Kůrka. "On topological dynamics of Turing machines". In: *Theoretical Computer Science* 174.1–2 (1997), pp. 203–216 (cit. on pp. 4, 5).
DOI: 10.1016/S0304-3975(96)00025-4.

[LM18] Frédéric Le Roux and Kathryn Mann. "Strong distortion in transformation groups". In: *Bulletin of the London Mathematical Society* 50.1 (2018), pp. 46–62 (cit. on pp. 1, 29).
DOI: 10.1112/blms.12108.

[LM95] Douglas Lind and Brian Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1995. ISBN: 978-1108820288 (cit. on p. 31).
DOI: 10.1017/CBO9780511626302.

[Mey11] Tom Meyerovitch. "Growth-type invariants for $\mathbb{Z}^d$ subshifts of finite type and arithmetical classes of real numbers". In: *Inventiones Mathematicae* 184.3 (2011), pp. 567–589 (cit. on p. 43).
DOI: 10.1007/s00222-010-0296-1.

[Moo91] Cristopher Moore. "Generalized shifts: unpredictability and undecidability in dynamical systems". In: *Nonlinearity* 4.2 (1991), p. 199 (cit. on p. 4).

[Nav21] Andrés Navas. "(Un)distorted diffeomorphisms in different regularities". In: *Israel Journal of Mathematics* 244.2 (2021), pp. 727–741 (cit. on pp. 1, 29).
DOI: 10.1007/s11856-021-2188-z.

[Oll18] Nicolas Ollinger. "On Aperiodic Reversible Turing Machines (invited talk)". In: *Reversible Computation 10th International Conference (RC 2018)*. Vol. 11106. Lecture Notes in Computer Science. Springer, 2018, pp. 61–64 (cit. on pp. 4, 6).
DOI: 10.1007/978-3-319-99498-7_4.

[Ore51] Oystein Ore. "Some Remarks on Commutators". In: *Proceedings of the American Mathematical Society* 2.2 (1951), pp. 307–314 (cit. on pp. 21, 35).
DOI: 10.2307/2032506.

[Pen20] Mark Pengitore. "Residual finiteness and strict distortion of cyclic subgroups of solvable groups". In: *Journal of Algebra* 546 (2020), pp. 679–688 (cit. on pp. 1, 29).
DOI: 10.1016/j.jalgebra.2019.11.011.

[PS22] Ronnie Pavlov and Scott Schmieding. "Local finiteness and automorphism groups of low complexity subshifts". In: *Ergodic Theory and Dynamical Systems* (2022), pp. 1–22 (cit. on pp. 29, 44).
DOI: 10.1017/etds.2022.7.

[Rog75] Yuuri V. Rogozhin. "Undecidability of the immortality problem for Turing machines with three states". In: *Cybernetics* 11.1 (1975), pp. 46–49 (cit. on p. 4).
DOI: 10.1007/BF01069942.

[Sal18] Ville Salo. "A note on subgroups of automorphism groups of full shifts". In: *Ergodic Theory and Dynamical Systems* 38.4 (2018), pp. 1588–1600 (cit. on p. 43).
DOI: 10.1017/etds.2016.95.

[Sal20] Ville Salo. *Graph and wreath products of cellular automata*. 2020 (cit. on p. 29).
DOI: 10.48550/ARXIV.2012.10186.

[Sal22] Ville Salo. "Universal groups of cellular automata". In: *Colloquium Mathematicum* 169.1 (2022), pp. 39–77 (cit. on p. 17).
DOI: 10.4064/cm8368-9-2021.

[Sch20] Scott Schmieding. "Automorphisms of the shift: Lyapunov exponents, entropy, and the dimension representation". In: *Ergodic Theory and Dynamical Systems* 40.9 (2020), pp. 2552–2570 (cit. on p. 30).
DOI: 10.1017/etds.2019.9.

[ST12] Ville Salo and Ilkka Törmä. "Computational aspects of cellular automata on countable sofic shifts". In: *Mathematical Foundations of Computer Science 2012*. Vol. 7464. Lecture Notes in Computer Science. Springer, Heidelberg, 2012, pp. 777–788 (cit. on p. 43).
DOI: 10.1007/978-3-642-32589-2_67.