Received: 11 December 2024 | Revised: 26 November 2025 | Accepted: 19 March 2026

DOI: 10.1112/jlms.70538

RESEARCH ARTICLE

Journal of the London Mathematical Society

On the ET0L subgroup membership problem in bounded automata groups

Alex Bishop[1] | **Daniele D'Angeli**[2] | **Francesco Matucci**[3] | **Tatiana Nagnibeda**[1] | **Davide Perego**[1] | **Emanuele Rodaro**[4]

[1]Université de Genève, Geneva, Switzerland

[2]Università Niccolò Cusano, Rome, Italy

[3]Università di Milano–Bicocca, Milan, Italy

[4]Politecnico di Milano, Milan, Italy

Correspondence
Davide Perego, Université de Genève, Geneva, Switzerland.
Email: davide.perego@unige.ch

Funding information
NSF, Grant/Award Number: 200020-200400; Spanish Ministry of Science and Innovation; NextGenerationEU; Italian Ministry of University and Research

Abstract

We are interested in the subgroup membership problem in groups acting on rooted d-regular trees and a natural class of subgroups, the stabilisers of infinite rays emanating from the root. These rays, which can also be viewed as infinite words in the alphabet with d letters, form the boundary of the tree. Stabilisers of infinite rays are not finitely generated in general, but if the ray is computable, the membership problem is well-posed and solvable. The main result of the paper is that, for bounded automata groups, the membership problem in the stabiliser of any ray that is eventually periodic as an infinite word, forms an ET0L language that is constructable. The result is optimal in the sense that, in general, the membership problem for the stabiliser of an infinite ray in a bounded automata group cannot be context-free. As an application, we give a recursive formula for the associated generating function, also known as the Green function, on the corresponding infinite Schreier graph.

MSC 2020
20E08 (primary), 05C25, 20F10, 68Q42, 68Q45 (secondary)

1 | INTRODUCTION

In an influential paper in 1911 [11], Max Dehn formulated three decision problems for finitely generated groups, the most famous of them being the *word problem*. For a finitely generated group G with a finite symmetric generating set X, the word problem asks if we can decide, given a word $w \in X^*$ in the free monoid over the alphabet X, whether $\overline{w}$, the natural projection of the word w to the group is the trivial element (in other words, belongs to the trivial subgroup). A natural extension of this problem is the *subgroup membership problem*, which asks, given a word $w \in X^*$ and a description of a subgroup $H \leqslant G$, whether $\overline{w}$ is an element of H. For G, X and $H \leqslant G$, denote by $\mathsf{WP}(G, X, H)$ the set of all words $w \in X^*$ for which $\overline{w}$ is an element of the subgroup H. We can then ask for which classes of finitely generated groups and subgroups the membership to $\mathsf{WP}(G, X, H)$ is uniformly computationally decidable. Of course, for the problem to be well-posed one needs the subgroups in question to have a computable description. The most popular case is to consider the membership problem in finitely generated subgroups, which, given finitely many elements $g_1, g_2, \dots, g_k \in G$ and a word $w \in X^*$, asks whether $\overline{w}$ belongs to the subgroup $H = \langle g_1, g_2, \dots, g_k \rangle$. Decidability of the subgroup membership problem (mostly for finitely generated subgroups) for various classes of groups has been studied over the years, and we refer the reader to the recent survey [17] for an excellent account of the state-of-the-art of the subject.

In this paper, we focus on one interesting aspect of the subgroup membership problem, that is, to describe the *formal language* $\mathsf{WP}(G, X, H)$ for given G, X and H. The word problem $\mathsf{WP}(G, X)$ can be thought of as the language formed by the words read along the closed paths in the Cayley graph of (G, X) where the edges are oriented and labelled by letters from X and the paths are based at the vertex representing the identity element. Similarly, the subgroup membership problem $\mathsf{WP}(G, X, H)$ consists of the words read along the closed paths in the Schreier graph of (G, X, H) based at the vertex representing the trivial coset H in the Schreier graph. Such languages have been studied extensively for the case when H is trivial (the word problem), but practically nothing is known for the case of non-trivial H. Anisimov proved in [2] that the word problem constitutes a regular language if and only if the group is finite, and this result readily generalises to arbitrary H: the language $\mathsf{WP}(G, X, H)$ is regular if and only if H is a subgroup of finite index [26, Proposition 6.1]. The famous theorem of Muller and Schupp [19] tells us that the word problem is a context-free language if and only if the group is virtually free.

Regular and context-free languages constitute two smallest classes in Chomsky's hierarchy of formal languages. The next class is that of context-sensitive languages, but in recent years other, intermediate classes came into play. For example, the *ET0L languages* introduced by Rozenberg in his 1973 paper [24] recently became popular in geometric group theory (see, e.g., [5, 8, 9, 12–14]). Recall that the regular (respectively, context-free) languages can be characterised as those that are recognised by finite-state (respectively, pushdown) automata. Analogously, ET0L languages are exactly those that can be recognised by a check-stack push-down automaton [31].

It is conjectured [9, Conjecture 8.1] that a group has an ET0L word problem if and only if it is virtually free. The main aim of this paper is to present a family of groups and subgroups where the class of subgroup membership problem is exactly ET0L.

We consider *groups generated by automorphisms of regular rooted trees*. For a fixed integer $d \geqslant 2$, we write $\mathcal{T}_d$ for the d-regular rooted tree. The vertices of $\mathcal{T}_d$ can be identified with words in the free monoid C^* in an alphabet $C = \{c_1, c_2, \dots, c_d\}$ with d letters. The one-sided infinite words in the alphabet C represent infinite rays emanating from the root that form the boundary $\partial\mathcal{T}_d$ of the tree. Let $G \leqslant \mathrm{Aut}(\mathcal{T}_d)$ be a finitely generated group of automorphisms of $\mathcal{T}_d$. Its action

on the tree by automorphisms extends by continuity to an action on the boundary of the tree by homeomorphisms.

Among the subgroups of G, a special role is played by *point stabilisers* for this action. The stabilisers of the tree vertices are subgroups of finite index and hence the subgroup membership problem in them is a regular language. From now on we concentrate on the membership problem in stabilisers of infinite rays. We denote by $\mathrm{Stab}(\eta)$ the stabiliser of the infinite ray $\eta = c_{i_1}c_{i_2}c_{i_3}\cdots \in C^\omega$, that is,

$$\mathrm{Stab}(\eta) = \bigcap_{k=1}^{\infty} \mathrm{Stab}(c_{i_1}c_{i_2}\cdots c_{i_k}) \qquad (\diamond)$$

where each $\mathrm{Stab}(c_{i_1}c_{i_2}\cdots c_{i_k})$ is the stabiliser in G of the vertex of the tree $\mathcal{T}_d$ corresponding to the word $c_{i_1}c_{i_2}\cdots c_{i_k}$.

We focus on *automaton automorphisms* of $\mathcal{T}_d$ (see Definition 2.1) which can be completely described by a finite amount of data.

This allows us to study the subgroup membership problem uniformly over the class of groups generated by finitely many automata automorphisms. This is a very interesting class of groups that includes many important examples, such as groups of intermediate growth, infinite torsion groups, non-elementary amenable groups, and more [6, 20].

We now turn to the subgroup membership problem to stabilisers of infinite rays in finitely generated automata groups. These stabiliser subgroups are not finitely generated in many interesting examples, but we can use the infinite ray η as an input of our algorithmic problem, and we require η to be computable.

To find groups and subgroups with ET0L subgroup membership problem, we further specialise to groups generated by *bounded automaton automorphisms*[†] (see Definition 2.3) and stabilisers of rays that are *eventually periodic*, that is, of the form $\eta = ab^\omega$ with $a, b \in C^*$.

In the main result of the paper, Theorem 4.1, we show that given a finite set of bounded automaton automorphisms X of the rooted tree $\mathcal{T}_d = C^*$, and words $a, b \in C^*$, the subgroup membership problem $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$, with $G = \langle X\rangle$ and $\eta = ab^\omega$, is an ET0L language as described in Section 3. In particular, we show that we can effectively compute a description of such an ET0L language by an *unambiguous limiting ET0L grammar* (see Definition 3.5). Describing an ET0L language in this way then enables us to apply Theorem 3.12 to find a description of its generating function.

Theorem 4.1. *Suppose that we are given a finite symmetric set X of bounded automaton automorphisms acting on the tree $\mathcal{T}_d = C^*$, and words $a, b \in C^*$. Then, we can effectively compute, uniformly over all X, a and b, an ET0L grammar which generates the language $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ with $G = \langle X\rangle$ and $\eta = ab^\omega$, and an ET0L grammar which generates the complement of this language, that is, $X^* \setminus \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$. Moreover, in both cases the grammars are unambiguous limiting.*

In language theory, a frequent question is where a specific class of formal languages fits within the Chomsky hierarchy. This hierarchy is a way to classify formal grammars and the languages they generate based on their expressive power. Formally, expressive power can be understood as the ability of the grammar to enforce increasingly complex dependencies between symbols

[†] Remark: given an automaton automorphism, it is computable to check if it is bounded.

in a string from local constraints in regular languages, to nested structures in context-free languages, and ultimately to arbitrary computable relations in recursively enumerable languages. It is well-known that the class of ET0L languages lies strictly between the families of context-free and context-sensitive languages (see [24, Theorem 19]). It is hence natural to ask whether our main result can be strengthened and whether the membership problem into stabilisers of infinite periodic rays belongs in fact to the class of context-free languages. In Section 5, we work with Schreier graphs of the corresponding subgroups to provide obstructions to the membership problem being context-free. In particular, we show that any stabiliser of an infinite ray η in the first Grigorchuk group and other key examples of bounded automata groups has a non-context-free language $\mathrm{WP}(G, X, \mathrm{Stab}(\eta))$. This demonstrates that Theorem 4.1 cannot be improved to the class of context-free languages.

The fact that the membership problems described in Theorem 4.1 are unambiguous limiting ET0L languages implies that their generating functions are computable. More precisely, we compute a recurrence relation for the generating functions of such unambiguous limiting ET0L languages in Theorem 3.12. For a background on the notation used in Theorem 3.12, see Section 3.3.

Theorem 3.12. *Let $L \subseteq \Sigma^*$ be an unambiguous limiting ET0L language. Then, it is computable to find a description of the generating function of L as*

$$f(z) = g(r_1(z), r_2(z), \dots, r_k(z))$$

where each $r_i(z) \in \mathbb{N}[[z]]$ is a rational power series, and $g(x_1, x_2, \dots, x_k)$ is a formal power series defined as

$$g(x_1, x_2, \dots, x_k) = \lim_{n\to\infty} g_n(x_1, x_2, \dots, x_k)$$

where $g_0(x_1, x_2, \dots, x_k) \in \mathbb{N}[[x_1, \dots, x_k]]$ is a rational power series, and

$$g_{n+1}(x_1, x_2, \dots, x_k) = g_n(q_1(x_1, x_2, \dots, x_k), q_2(x_1, x_2, \dots, x_k), \dots, q_k(x_1, x_2, \dots, x_k))$$

for each $n \geqslant 0$ where each $q_i(x_1, \dots, x_k) \in \mathbb{N}[[x_1, \dots, x_k]]$ is a rational series depending on g_n. In the above k is a constant that depends on the grammar.

Suppose that X is a finite symmetric set of automaton automorphisms which generates the group $G = \langle X \rangle$, and that $\eta \in \partial\mathcal{T}_d$, then the *Schreier graph* Γ_η of the subgroup $\mathrm{Stab}(\eta)$ is formed by the vertex set $\eta \cdot G = \{\eta \cdot g \mid g \in G\}$ with a labelled edge $\zeta \to^x (\zeta \cdot x)$ for each vertex ζ and each generator $x \in X$.

Theorem 3.12 can be used to calculate the generating functions of closed walks on such Schreier graphs. Indeed, Theorem 3.12 allows us to find a recurrence for the generating function $f(z)$ of the language $\mathrm{WP}(G, X, \mathrm{Stab}(\eta))$, as in Theorem 4.1. As mentioned above, $\mathrm{WP}(G, X, \mathrm{Stab}(\eta))$ consists exactly of all words read along the paths in Γ_η that begin and end at the vertex η. This generating function is closely related to the Green function of the simple random walk on Γ_η:

$$G(z) = f(z/|X|).$$

Recall that, for a random walk on the graph with the starting point η, its Green function is defined as

$$G(z) = \sum_{n=0}^{\infty} p^{(n)}(\eta, \eta)\, z^n$$

where $p^{(n)}(\eta, \eta)$ is the probability for the random walk to come back to η after exactly n steps. The random walk is called simple if its transition probabilities are uniform on neighbouring vertices. Our Theorem 3.12 can be extended to calculate the Green function for the more general case of random walks with non-uniform transition probabilities. Besides being an important characteristic of the random walk, the Green function and its complexity is valuable in particular in the study of spectra of graphs (see, e.g., [18]).

A corollary to Theorem 4.1 is that the membership problem to $\mathrm{Stab}(G, X, \mathrm{Stab}(\eta))$ is decidable uniformly over all bounded automata group G and uniformly over all eventually periodic rays η. Indeed, the membership to a fixed ET0L language is known to be decidable in space complexity $O(n)$ and time complexity $O(n^2)$ where n is the length of the input word (see [31, Lemma 2.1]); and it is possible to modify this procedure to make it uniform. However, uniform decidability of the subgroup membership problem for stabilisers of infinite rays follows from simpler arguments. In Propositions 2.2 and 2.10, we address this decision problem where η is a computable ray which is periodic or non-eventually periodic, respectively. The non-periodic case leads to a so-called *promise problem*, and in Proposition 2.11 we see that it cannot be extended to a decision procedure. We thank the anonymous reviewer for their suggested proof sketch of Proposition 2.2 which we give in Section 2.

Proposition 2.2. *Suppose we are given as input*

(1) *a finite symmetric set of automaton automorphisms X of the tree $\mathcal{T}_d = C^*$;*
(2) *a word $w \in X^*$ over the generating set X; and*
(3) *two words $a, b \in C^*$ with $|b| \geqslant 1$.*

Then it is computationally decidable if $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(ab^\omega))$. That is, the subgroup membership problem is solvable uniformly over all bounded automata groups $G = \langle X \rangle$, and uniformly over all eventually periodic rays $\eta = ab^\omega$.

We then treat the case of not eventually periodic computable rays and show that membership to $\mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$ is a so-called *promise problem*, solvable uniformly over all finite sets X of bounded automaton automorphisms and computable rays η, as long as we have a promise that the given ray is not eventually periodic.

This promise is necessary, as we see in Proposition 2.11.

Proposition 2.10. *Suppose we are given as input*

(1) *a finite symmetric set of bounded automaton automorphisms X of the tree $\mathcal{T}_d = C^*$;*
(2) *a word $w \in X^*$ over the generating set X; and*
(3) *a Turing machine T which outputs an infinite ray $\eta \in C^\omega$.*

Then, given the promise that T does not generate an eventually periodic ray, it is computationally decidable if $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$. That is, membership to $\mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$ is decidable as a

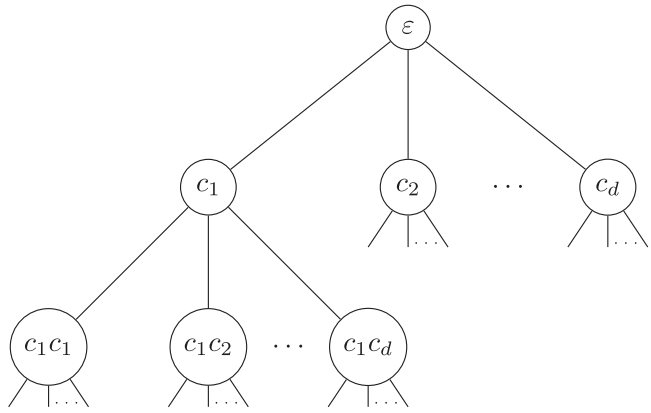

FIGURE 1 A labelling of the vertices of $\mathcal{T}_d$.

promise problem uniformly for all bounded automata groups and uniformly for all non-eventually periodic computable rays.

It is interesting to note that one cannot remove the promise from Proposition 2.10, in fact, if one attempts to do so then the problem is no longer computable, as shown in the following proposition.

Proposition 2.11. *There is no Turing machine which can take as input*

(1) *a finite symmetric set of bounded automaton automorphisms X of the tree $\mathcal{T}_d = C^*$;*
(2) *a word $w \in X^*$ over the generating set X; and*
(3) *a Turing machine T which outputs an infinite ray $\eta \in C^*$;*

then decide if $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$. In particular, this means that the computation as described in Proposition 2.10 cannot be generalised to a decision procedure, that is, the 'promise' in Proposition 2.10 is necessary.

The paper is organised as follows. Sections 2 and 3 are devoted to a general introduction to bounded automata groups and ET0L languages, but also serve to provide useful lemmas and definitions needed in the rest of the paper. Section 2 also contains the proofs of Propositions 2.2, 2.10 and 2.11. The proof of Theorem 4.1 is entirely contained in Section 4, while Section 5 gives criteria that can be used to show that the language of stabilisers is not context-free. Lastly, Section 6 contains a list of open questions and topics for future research.

2 | BOUNDED AUTOMATA GROUPS

As in the introduction, we write $\mathcal{T}_d$ for the d-regular rooted tree. We identify the vertices with the words in C^* where $C = \{c_1, c_2, \ldots, c_d\}$, with the root labelled by the empty word $\varepsilon \in C^*$. See Figure 1 for a depiction of this tree.

We write $\mathrm{Aut}(\mathcal{T}_d)$ for the group of automorphisms of $\mathcal{T}_d$. Every automorphism $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ fixes the root and preserves the levels of the tree. In fact, $\mathrm{Aut}(\mathcal{T}_d) = \mathrm{Aut}(\mathcal{T}_d) \wr \mathrm{Sym}(C)$ where $\mathrm{Sym}(C)$ is the symmetric group on the set C. That is, each automorphism $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ can be uniquely

written in the form $\alpha = (\alpha'_1, \alpha'_2, \dots, \alpha'_d) \cdot s$ where each $\alpha'_i \in \mathrm{Aut}(\mathcal{T}_d)$ is an automorphism of the subtree rooted at c_i (which is isomorphic to $\mathcal{T}_d$), and $s \in \mathrm{Sym}(C)$ is a permutation of the subtrees rooted in the vertices of the first level. The automorphism α'_i, with $i = 1, 2, \dots, d$, is called the *section of* α at c_i, and will be denoted $\alpha@c_i$. Then the *section* $\alpha@v$ at an arbitrary vertex $v = c_{i_1} c_{i_2} \cdots c_{i_m} \in C^*$ is defined recursively as

$$\alpha@v = \left(\alpha@c_{i_1}\right)@c_{i_2} c_{i_3} \cdots c_{i_m}.$$

That is, $\alpha@v$ is the action that the element α has on the subtree rooted at v.

Definition 2.1. An automorphism $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ is an *automaton automorphism* if there exists a finite set $A_\alpha \subset \mathrm{Aut}(\mathcal{T}_d)$ such that $\alpha@v \in A_\alpha$ for each $v \in C^*$. The set of all automaton automorphisms forms a group $\mathrm{AAut}(\mathcal{T}_d)$. A group $G \leqslant \mathrm{Aut}(\mathcal{T}_d)$ is called an *automata group* if $G \leqslant \mathrm{AAut}(\mathcal{T}_d)$.

In the literature, the class of automaton automorphisms is usually introduced by first defining a computational model known as a finite-state automaton, for example, see [20, Definition 1.3.1]. Our definition is equivalent, in particular, a finite set of states for an automaton representing an automaton automorphism α is given by the set A_α from Definition 2.1. Automata groups are often assumed to be self-similar (or state-closed). We do not make this assumption until Section 5 when it will be explicitly specified.

From the definition of an automaton automorphism, it is clear that it is determined by a finite amount of data. Therefore, any such automorphism can be encoded and provided as input to a Turing machine. Moreover, it is computable to check if a given automaton is the identity, and composition of such automorphisms is also computable, see [20, Section 1.3.5] for more details.

We will be interested here in the subgroup membership problem in automata groups. As explained in the introduction, an important family of subgroups in a group of automorphisms of a regular rooted tree is formed by the stabilisers of the vertices of the tree and of the elements of the boundary of the tree (also known as infinite rays emanating from the root). We noted in the introduction that the subgroup membership problem in the stabiliser of a vertex of the tree is a regular language, and from now on we only concentrate on the problem of membership in the stabilisers of infinite rays. At this point, we can show that for an automata group, the subgroup membership problem is decidable for stabilisers of eventually periodic rays. We thank the anonymous reviewer for suggesting to us a sketch of the proof of the following Proposition.

Proposition 2.2. *Suppose we are given as input*

(1) *a finite symmetric set of automaton automorphisms X of the tree $\mathcal{T}_d = C^*$;*
(2) *a word $w \in X^*$ over the generating set X; and*
(3) *two words $a, b \in C^*$ with $|b| \geqslant 1$.*

Then it is computationally decidable if $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(ab^\omega))$. That is, the subgroup membership problem is solvable uniformly over all bounded automata groups $G = \langle X \rangle$, and uniformly over all eventually periodic rays $\eta = ab^\omega$.

Proof. Suppose that we are given some finite set X and segments $a, b \in C^*$. Suppose also that we are given a word $w \in X^*$. Then, we can compute a description of the corresponding element

$\overline{w} \in \mathrm{AAut}(\mathcal{T}_d)$. In particular, suppose that $A_{\overline{w}}$ is the finite set of automorphisms as in Definition 2.1. Let $K = |A_{\overline{w}}|$. As we have a description of the automaton automorphism $\overline{w}$, we can compute the vertex $v = (a\, b^{K+1}) \cdot \overline{w}$. In the remainder of this proof, we show that $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$ if and only if $v = a\, b^{K+1}$.

Suppose that $v \neq a\, b^{K+1}$, then the action of $\overline{w}$ is non-trivial on some prefix of η and thus is non-trivial on η. From this, we conclude that $w \notin \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$ as required.

Now suppose that $v = a\, b^{K+1}$, that is, that $\overline{w}$ has trivial action on the vertex $a\, b^{K+1}$. By the pigeonhole principle, we then see that there must exist two distinct values $k_1, k_2 \in \{0, 1, \ldots, K\}$ with $k_1 > k_2$ such that

$$\beta := \overline{w} @ \left(a\, b^{k_1}\right) = \overline{w} @ \left(a\, b^{k_2}\right),$$

where $b^{k_1 - k_2} \cdot \beta = b^{k_1 - k_2}$. Thus, from the definition of the automaton automorphisms we see that

$$\eta \cdot \overline{w} = (ab^{\omega}) \cdot \overline{w} = \left(ab_2^k\right)(b^{k_1 - k_2})^{\omega} = \eta.$$

That is, $\overline{w}$ has trivial action on the ray η. □

From now on, we will be interested in $\mathcal{T}_d$ by *bounded automaton automorphisms* defined as follows.

Definition 2.3. We say that an automorphism $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ is *bounded* [28] if there exists some constant N_α such that

$$\#\{v \in C^* \mid \alpha @ v \neq 1 \text{ and } |v| = k\} < N_\alpha$$

for every positive integer k. The set of all bounded automaton automorphisms forms a group which we denote $\mathcal{B}(\mathcal{T}_d) < \mathrm{AAut}(\mathcal{T}_d)$. A group G is called a *bounded automata group* if $G \leqslant \mathcal{B}(\mathcal{T}_d)$.

In [28], Sidki considered two classes of bounded automaton automorphisms, known as *finitary* and *directed* automaton automorphisms, and showed that they form a generating set for the group of bounded automaton automorphisms $\mathcal{B}(\mathcal{T}_d)$, see Proposition 2.7.

Definition 2.4. An automorphism $\phi \in \mathrm{Aut}(\mathcal{T}_d)$ is *finitary* if there exists a constant $N_\phi \in \mathbb{N}$ such that $\phi @ v = 1$ for each $v \in C^*$ with $|v| \geqslant N_\phi$. The smallest constant for which this holds is the *depth* of the automorphism ϕ, denoted as $\mathrm{depth}(\phi)$.

Finitary automorphisms form a subgroup of $\mathcal{B}(\mathcal{T}_d)$ which we denote as $\mathrm{Fin}(\mathcal{T}_d)$. Examples of finitary automorphisms are given in Figure 2

For any automorphism $\phi \in \mathrm{Aut}(\mathcal{T}_d)$, finitary or not, we will also need the notion of its ' directional depth' with respect to a vertex of the tree or with respect to an infinite ray, that we now introduce.

Definition 2.5. The *directional depth* of an automorphism with respect to a word, finite or infinite, in the alphabet C is given by the function $\mathrm{DDepth} : (C^* \cup C^{\omega}) \times \mathrm{Aut}(\mathcal{T}_d) \to \mathbb{N} \cup \{\infty\}$ defined

FIGURE 2 Examples of finitary automorphisms $a, b \in \mathrm{Fin}(\mathcal{T}_d)$.

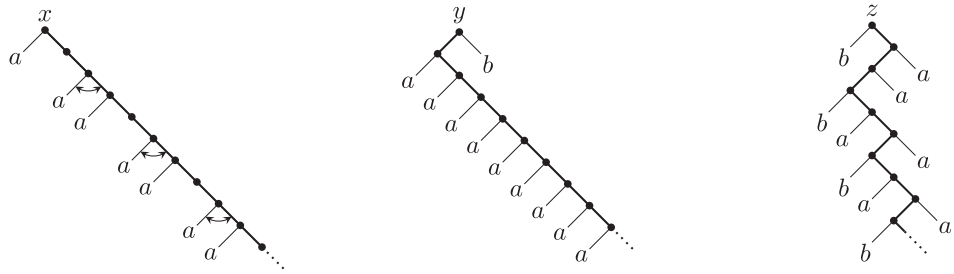

FIGURE 3 Examples of directed automorphisms $x, y, z \in \mathrm{Dir}(\mathcal{T}_2)$.

as

$$\mathrm{DDepth}(\zeta, x) = \begin{cases} \min\{|v| \mid v \text{ is a prefix of } \zeta \text{ with } x@v = 1\} \\ \infty \text{ if } x@v \neq 1 \text{ for each prefix } v \text{ of } \zeta. \end{cases}$$

Let us now turn to directed automorphisms.

Definition 2.6. A bounded automaton automorphism $\delta \in \mathrm{Aut}(\mathcal{T}_d)$ is *directed* if there exists a unique infinite word $c_{i_1} c_{i_2} \dots c_{i_m} \dots$ such that $\delta@c_{i_1} c_{i_2} \dots c_{i_m} \neq 1$ for all m. Such word is called *spine* and denote by $\mathrm{spine}(\delta)$.

We denote the set of all directed automaton automorphisms as $\mathrm{Dir}(\mathcal{T}_d)$. See Figure 3 for some examples (in these examples, a and b are as in Figure 2).

The composition of two elements in $\mathrm{Dir}(\mathcal{T}_d)$ might not be in $\mathrm{Dir}(\mathcal{T}_d)$. However, the resulting automorphism has at most two infinite rays with the same behaviour of the spines of the directed automorphisms. Indeed, if $\delta, \delta' \in \mathrm{Dir}(\mathcal{T}_d)$, then either $\mathrm{spine}(\delta) \cdot \delta = \mathrm{spine}(\delta')$ and $\delta\delta'$ is still a directed automorphism with $\mathrm{spine}(\delta\delta') = \mathrm{spine}(\delta)$, or $\delta\delta'$ has two infinite rays $\mathrm{spine}(\delta)$ and $\mathrm{spine}(\delta') \cdot \delta^{-1}$ such that the restrictions on each prefix is non-trivial (note that δ^{-1} is acting as a finitary automorphism on δ'). The same holds more generally for a product of directed automorphisms.

Proposition 2.7 [28, Proposition 16]. *The group $\mathcal{B}(\mathcal{T}_d)$ of bounded automaton automorphisms is generated by* $\mathrm{Fin}(\mathcal{T}_d)$ *together with* $\mathrm{Dir}(\mathcal{T}_d)$.

The proof of Proposition 16 in [28] is constructive, that is, given a bounded automaton automorphism α, it is computable to find a finite decomposition $\alpha = s_1 s_2 \cdots s_k$ where each $s_i \in \mathrm{Fin}(\mathcal{T}_d) \cup \mathrm{Dir}(\mathcal{T}_d)$. This follows since it is composition and equality is computable in the set of

automaton automorphisms. In particular, given a bounded automaton automorphism, α, one can non-deterministically choose such a decomposition $s_1 s_2 \cdots s_k$, where each $s_i \in \mathrm{Fin}(\mathcal{T}_d) \cup \mathrm{Dir}(\mathcal{T}_d)$, then verify that it presents the same automorphism as α.

Proposition 2.8 [5, Lemma 3, p. 87]. *The spine,* $\mathrm{spine}(\delta) \in C^\omega$*, of a directed automaton automorphism,* $\delta \in \mathrm{Dir}(\mathcal{T}_d)$*, is eventually periodic, that is, there are words* $u = u_1 u_2 \cdots u_s \in C^*$ *and* $v = v_1 v_2 \cdots v_t \in C^*$ *with* $v \neq \varepsilon$*, called the* initial *and* periodic *segment, respectively, for which* $\mathrm{spine}(\delta) = uv^\omega$ *and*

$$\delta@uv^k v_1 v_2 \cdots v_j = \delta@uv_1 v_2 \cdots v_j$$

for each $k, j \in \mathbb{N}$ *with* $0 \leqslant j \leqslant t$.

This generalises to any element of $\mathcal{B}(\mathcal{T}_d)$. Since the product of a directed automorphism and a finitary automorphism is directed and by the discussion right after Definition 2.6 on the product of directed automorphism, by Proposition 2.7 is clear that a bounded automaton automorphism has finitely many infinite rays such that on each prefix the restriction is non-trivial. Moreover, the action of an element acting like a finitary automorphism on an infinite word only changes a finite prefix, and by the same discussion as above, we have that the infinite rays of a bounded automaton automorphism are all eventually periodic.

We are now ready to prove the following lemma which is used to simplify the proof of Theorem 4.1.

Lemma 2.9. *Let* $G < \mathcal{B}(\mathcal{T}_d)$ *be a finitely generated bounded automata group. There exists a bounded automata group* H *with a finite symmetric generating set* $S \subset \mathrm{Fin}(\mathcal{T}_d) \cup \mathrm{Dir}(\mathcal{T}_d)$ *such that* G *is a subgroup of* H*. Moreover, such a generating set* S *is effectively constructable from a finite set of bounded automaton automorphisms* X *where* $G = \langle X \rangle$.

Proof. Let X be a finite generating set for the group G. From Proposition 2.7, we see that for each $x \in G$, in particular, for each $x \in X$, there is a word $x = w_{x,1} w_{x,2} \cdots w_{x,k(x)}$ with each $w_{x,i} \in \mathrm{Fin}(\mathcal{T}_d) \cup \mathrm{Dir}(\mathcal{T}_d)$. Define a symmetric generating set

$$S = \{w_{x,j}, (w_{x,j})^{-1} \mid x \in X, j \in \{1, 2, \ldots, k(x)\}\}.$$

From the definition of finitary and directed automaton automorphisms, we see that $S \subset \mathrm{Fin}(\mathcal{T}_d) \cup \mathrm{Dir}(\mathcal{T}_d)$, as desired. Moreover, the group generated by S contains G as a subgroup. It follows from the observation after Proposition 2.7 that the finite set S is computable from X. □

In Proposition 2.2, we showed that the subgroup membership problem for the stabiliser subgroup of an eventually periodic ray is computable for arbitrary finitely generating sets of automaton automorphisms. In the case of bounded automata automorphisms, we are able to show the following companion property for non-eventually periodic computable rays. However, it is a promise problem, moreover, we show in Proposition 2.11 below that it cannot be turned into a proper decision problem.

Proposition 2.10. *Suppose we are given as input*

(1) *a finite symmetric set of bounded automaton automorphisms X of the tree $\mathcal{T}_d = C^*$;*
(2) *a word $w \in X^*$ over the generating set X; and*
(3) *a Turing machine T which outputs an infinite ray $\eta \in C^\omega$.*

Then, given the promise that T does not generate an eventually periodic ray, it is computationally decidable if $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$. That is, membership to $\mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$ is decidable as a promise problem uniformly for all bounded automata groups and uniformly for all non-eventually periodic computable rays.

Proof. We begin by computing an automaton automorphism for the action of $\overline{w}$.

Since the infinite rays of a bounded automaton automorphism is eventually periodic (see Proposition 2.8 and the discussion right after it), we then see that the ray η, as described by T, must eventually leave this finite set of rays. Thus, we see that $\overline{w}$ only performs an action on a finite prefix of η. Moreover, the length of this prefix is computable from the description of the automaton for $\overline{w}$ and prefixes of η. From these observations, it follows immediately that $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$ is decidable. □

Proposition 2.11. *There is no Turing machine which can take as input*

(1) *a finite symmetric set of bounded automaton automorphisms X of the tree $\mathcal{T}_d = C^*$;*
(2) *a word $w \in X^*$ over the generating set X, and*
(3) *a Turing machine T which outputs an infinite ray $\eta \in C^*$;*

then decide if $w \in \mathsf{WP}(\langle X \rangle, X, \mathrm{Stab}(\eta))$. In particular, this means that the computation as described in Proposition 2.10 cannot be generalised to a decision procedure, that is, the 'promise' in Proposition 2.10 is necessary.

Proof. We begin by introducing a decision problem which we call *Periodicity*:

Input: A Turing-machine-based description of an infinite ray $\eta \in \{0,1\}^\omega$.
Question: Is $\eta = 1^\omega$?

Periodicity is essentially a reformulation of the halting problem. For completeness, let us outline the reduction. Given a Turing machine T, construct a machine P that simulates T and, at each step of the simulation, outputs the letter 1. If T ever halts then P switches and outputs only 0's from that point onward, thus producing the sequence $1^k 0^\omega$ for some $k \in \mathbb{N}$. Hence, P outputs the infinite sequence 1^ω if and only if T does not halt. Therefore, if the problem *Periodicity* were decidable, then the halting problem would also be decidable, a contradiction.

We now reduce *Periodicity* to the problem in the statement. Consider the infinite dihedral group $\mathcal{D}$ (see Figure 5), a well-known bounded automaton group generated by a and b. It is straightforward to check that for an infinite ray η one has $\eta \cdot b = \eta$ if and only if $\eta = 1^\omega$. Suppose by contradiction that our problem is decidable. Then there exists a Turing machine M that, given a bounded automaton group G, a Turing machine T_η producing the infinite ray η, and a word $w \in X^*$ over the generating set X of G, decides whether $w \in \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$. Applied to the input $(\mathcal{D}, T_\eta, b)$, the machine M would decide whether $b \in \mathsf{WP}(\mathcal{D}, \{a, b\}, \mathrm{Stab}(\eta))$, that is, whether $\eta \cdot b = \eta$, which is equivalent to deciding whether $\eta = 1^\omega$. Thus, M would solve *Periodicity*, which we have shown to be undecidable. This contradiction shows that such a machine M cannot exist. □

3 | ET0L LANGUAGES

In this section, we define and provide a background on the family of *Extended Tabled 0-interaction Lindenmayer* (*ET0L*) languages (see Definition 3.2) which was introduced and studied by Rozenberg [24]. We begin by defining the ET0L languages in terms of a class of formal grammars. We conclude this section by studying a particular subclass of ET0L language in Section 3.1 and show in Section 3.3 that ET0L languages from this class have generating functions which we can specify using equations of a particular form.

Below, we give a definition of ET0L languages which is due to Asveld [3]. In particular, the definition we use in this paper is what Asveld refers to as a (REG, REGITER) grammar (cf. the definitions in [3, pp. 253–254]). The proof that this is equivalent to the definition given by Rozenberg in [24] follows from [3, Theorem 2.1], and [21, Theorems 2 and 3]. (Note that RC-Part ET0L in [21] has the same definition as (REGITER) in [3].)

An *ET0L grammar* is a type of replacement system which has both a terminal alphabet Σ and a disjoint non-terminal alphabet V. In particular, our grammar begins with an initial symbol $S \in V$. We then perform a sequence of allowable replacements to this symbol until we have a word which consists of only letters in Σ. Such a word is then said to be generated by the grammar. Allowable replacements are given by *tables*, defined as follows.

Definition 3.1. A *table* is a function of the form $\tau : \Sigma \cup V \to \mathrm{Reg}(\Sigma \cup V)$ where $\mathrm{Reg}(\Sigma \cup V)$ denotes the family of regular languages over the alphabet $\Sigma \cup V$ and $\tau(\sigma) = \{\sigma\}$ for each $\sigma \in \Sigma$.

Since the elements of Σ are fixed, we do not specify them when we explicitly provide a table. Suppose that $\tau : \Sigma \cup V \to \mathrm{Reg}(\Sigma \cup V)$ is a table as defined above. Then we write $w \to^{\tau} w'$ for each word $w = w_1 w_2 \cdots w_m \in (\Sigma \cup V)^*$ and each word $w' = w'_1 w'_2 \cdots w'_m$ where each w'_i belongs to the regular language $\tau(w_i)$. For tables $\tau_1, \tau_2, \ldots, \tau_k$, we write $w \to^{\tau_1\tau_2\cdots\tau_k} w'$ if there are words $w_1, w_2, \ldots, w_{k+1} \in (\Sigma \cup V)^*$ with $w_1 = w$, $w_{k+1} = w'$ and $w_i \to^{\tau_i} w_{i+1}$ for each i.

For example, let $\Sigma = \{a, b\}$ and $V = \{S, A, B\}$, then

$$\alpha : \begin{cases} S \mapsto \{SS, S, AB\} \\ A \mapsto \{A\} \\ B \mapsto \{B\} \end{cases} \qquad \beta : \begin{cases} S \mapsto \{S\} \\ A \mapsto \{aA\} \\ B \mapsto \{bB\} \end{cases} \qquad \gamma : \begin{cases} S \mapsto \{S\} \\ A \mapsto \{\varepsilon\} \\ B \mapsto \{\varepsilon\} \end{cases} \tag{1}$$

are tables. We see that $w \to^{\alpha} w'$ where $w = SSSS$ and $w' = SABSSAB$.

We can now define ET0L grammars, as follows.

Definition 3.2. An *ET0L grammar* is a 5-tuple $E = (\Sigma, V, T, \mathcal{R}, S)$, where

(1) Σ is an alphabet of *terminals*;
(2) V is an alphabet of *non-terminals*;
(3) $T = \{\tau_1, \tau_2, \ldots, \tau_k\}$ is a finite set of *tables*;
(4) $\mathcal{R} \subseteq T^*$ is a regular language called the *rational control*; and
(5) $S \in V$ is the *start symbol*.

We then say that

$$L(E) = \{w \in \Sigma^* \mid S \to^{v} w \text{ for some } v \in \mathcal{R}\}$$

is the *ET0L language* generated by the grammar E

For example, let α, β and γ be as in (1), then the language that is produced by the grammar with rational control $\mathcal{R} = \alpha^*\beta^*\gamma$ is $\{(a^n b^n)^m \mid n, m \in \mathbb{N}\}$. It can then be shown, using the *pumping lemma* (see [29, Theorem 2.34]), that this language is not context-free. It is known that every context-free language is also ET0L (see the diagram in T28 in [25, p. 241]).

We now introduce some additional notation which will be used in the proof of Theorem 4.1. We prefer this notation as it matches the way in which we apply tables from left to right.

Notation 3.3. Suppose that $E = (\Sigma, V, T, \mathcal{R}, S)$ is an ET0L language, then for each word $w \in (\Sigma \cup V)^*$ and each sequence of tables $t \in T^*$, we write

$$w \cdot t = \{u \in (\Sigma \cup V)^* \mid w \to^{t} u\}$$

for the set of all words which can be obtained from w by applying t.

3.1 | Unambiguous limiting grammars

Each of the ET0L grammars that we construct in Theorem 4.1 has a particular form for which one can compute a description of their generating function (see Theorem 3.12). In this subsection, we give a description of this class of grammars. Later in this section, we prove some closure properties, and study the combinatorial complexity of this class. We begin by describing we mean for a grammar to be *unambiguous* as follows.

Similarly to context-free languages, we can define derivation trees for ET0L grammars. However, for the derivation tree from an ET0L language, we label each level of the tree to denote the table which is being applied. For example, consider the language of partitions given as

$$L = \{a^{n_1} b a^{n_2} b \cdots b a^{n_k} b \mid k \geqslant 1 \text{ and } n_1 \geqslant n_2 \geqslant \cdots \geqslant n_k \geqslant 1\}.$$

It was shown in [9] that this language is ET0L, in particular, it is generated by an ET0L language with non-terminals S and A, and tables

$$\alpha : \begin{cases} S \mapsto aAbS \\ A \mapsto A \end{cases} \qquad \beta : \begin{cases} S \mapsto S \\ A \mapsto aA \end{cases} \qquad \text{and} \qquad \gamma : \begin{cases} S \mapsto \varepsilon \\ A \mapsto \varepsilon \end{cases}$$

with rational control $\mathcal{R} = \{\alpha, \beta, \gamma\}^*$. Notice that the word a^2babab belongs to the language L. In particular, this word has a derivation tree labelled by $\alpha\beta\alpha\alpha\gamma$ as given in Figure 4. We obtain the word a^2babab from the tree given in Figure 4 by reading off the leaves from left to right.

We then say that a derivation tree as in Figure 4 is a derivation tree labelled by $\alpha\beta\alpha\alpha\gamma$ and that for each word w with $S \to^{\alpha\beta\alpha\alpha\gamma} w$, there is a derivation tree with the same labelling. Moreover, by considering other first 4 levels of Figure 4, we see that the word $aaAaAbS$ has a derivation tree labelled by $\alpha\beta\alpha\alpha$. That is, we allow our derivation trees to have non-terminals in their leaves. We now define unambiguous ET0L languages as follows.

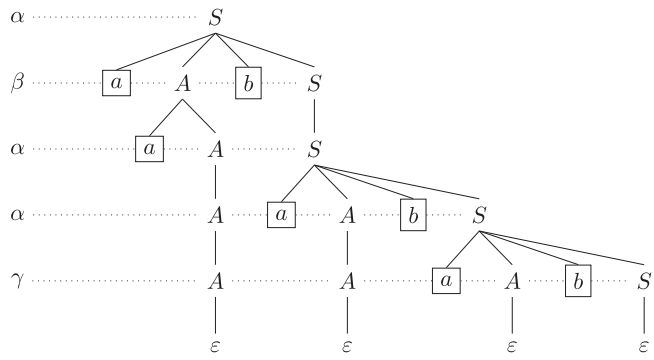

FIGURE 4 Derivation tree for a^2babab labelled by $\alpha\beta\alpha\alpha\gamma$.

Definition 3.4. Let E be an ET0L grammar, then we say that E is *unambiguous with respect to its rational control* (or simply *unambiguous*) if for every $r \in \mathcal{R}$, and every word $w \in (\Sigma \cup V)^*$, there is at most one derivation tree for w labelled by r.[†]

We now define the class of ET0L grammars, as in the proof of Theorem 4.1, as follows.

Definition 3.5. Let $E = (\Sigma, V, T, \mathcal{R}, S)$ be an ET0L grammar as in Definition 3.2, then we say that E is *limiting* if

(1) T contains 3 tables, that is, $T = \{\alpha, \beta, \gamma\}$;
(2) the rational control is given by $\mathcal{R} = \alpha\beta^*\gamma$;
(3) we have $\varepsilon \notin \beta(v)$ for each $v \in V$;
(4) we have $\gamma(v) \neq \emptyset$ for each $v \in V$;
(5) (limiting) if $S \to^{\rho_n} w$ for some $w \in (\Sigma \cup V)^*$ where $\rho_n = \alpha\beta^n\gamma$, then there exists some $K \geqslant n$ such that $S \to^{\rho_k} w$ for each $k \geqslant K$.

Requirements (3) and (4) are technical requirements which allow us to compute the generating function.

We then say that an ET0L grammar is *unambiguous limiting* if it satisfies the properties in Definitions 3.4 and 3.5. In the following subsection, we show that the class of unambiguous limiting grammars is closed under mappings by injective string transducers. This result is then used to prove Proposition 3.11 which is used to simplify the proof of our main theorem (i.e. Theorem 4.1).

3.2 | Closure under mapping by string transducer

We now show that an unambiguous limiting ET0L language is closed under mapping by a *string transducer*, also known as a *deterministic finite-state transducer*, or a *deterministic generalised sequential machine (deterministic gsm)*. We begin with the following definition.

Definition 3.6. A (deterministic) *string transducer* is a tuple $M = (\Gamma, \Sigma, Q, A, q_0, \delta)$ where

[†] Here we consider all words over terminals and non-terminals, not just words only in terminal letters.

- Γ and Σ are the *input* and *output alphabets*, respectively;
- Q is a finite set of *states*;
- $A \subseteq Q$ is a finite set of *accepting states*;
- $q_0 \in Q$ is the *initial state*; and
- $\delta : \Gamma \times Q \to \Sigma^* \times Q$ is a *transition function*.

Given a language $L \subseteq \Gamma^*$, we may then define the language $M(L) \subseteq \Sigma^*$ as

$$M(L) = \left\{ u_1 u_2 \cdots u_k \in \Sigma^* \;\middle|\; \begin{array}{r} \text{there exists some word } w = w_1 w_2 \cdots w_k \in L \subseteq \Gamma^* \\ \text{such that } \delta(w_i, q_{i-1}) = (u_i, q_i) \text{ for each } i \in \{1, 2, \dots, k\} \\ \text{where } q_0 \text{ is the initial state, and } q_1, q_2, \dots, q_k \in Q \text{ with } q_k \in A \end{array} \right\}.$$

We then say that $M(L)$ is the image of L under mapping by the string transducer M.

Let $M = (\Gamma, \Sigma, Q, A, q_0, \delta)$ be a string transducer. Then, for each pair of states $q, q' \in Q$, and words $w = w_1 w_2 \cdots w_k \in \Gamma^*$ and $w' \in \Sigma^*$, we write $q \to^{(w,w')} q'$ if there is a path from state q to q' which rewrites the word w to w'; that is, if there is a sequence of states $q_1, q_2, \dots, q_{k+1} \in Q$ such that

- $q = q_1$ and $q' = q_{k+1}$; and
- $\delta(w_i, q_i) = (u_i, q_{i+1})$ for each $i \in \{1, 2, \dots, k\}$ where $w' = u_1 u_2 \cdots u_k$.

We then see that the language $M(L)$ can be written as

$$M(L) = \{w' \in \Sigma^* \mid q_0 \to^{(w,w')} q \text{ where } w \in L \text{ and } q \in A\}$$

for each $L \subseteq \Gamma^*$.

Definition 3.7. We say that a string transducer $M = (\Gamma, \Sigma, Q, A, q_0, \delta)$ is *injective* if for each $w' \in \Sigma^*$, there is at most one word $w \in \Gamma^*$ such that $q_0 \to^{(w,w')} q$ with $q \in A$. That is, if we view M as a partial map from Γ^* to Σ^*, then it is injective in the usual sense.

Our main objective in this subsection is to prove the following proposition.

Proposition 3.8. *Suppose that we are given an unambiguous limiting ET0L grammar for some language $L \subseteq \Sigma^*$, and an injective string transducer $M = (\Sigma, \Gamma, Q, A, q_0, \delta)$. Then, it is computable to find an unambiguous limiting ET0L grammar for the language $L' = M(L)$.*

Proof. Let $E = (\Sigma, V, T, \mathcal{R}, S)$ be an unambiguous limiting ET0L grammar for the language L, and let $M = (\Sigma, \Gamma, Q, A, q_0, \delta)$ be a string transducer. We are constructing an ET0L grammar $E' = (\Sigma, V', \mathcal{R}', S')$ which recognises the language $L' = M(L)$. The grammar E' is obtained by annotating the non-terminals in E with paths in the string transducer M. We then modify the grammar E' and show that it is unambiguous limiting. We begin our construction by listing out the non-terminals of E' as follows.

Non-terminals:

We begin by introducing the initial non-terminals of E' which we write as $S' \in V$. For each non-terminal $v \in V$, and each pair of states $q, q' \in Q$, we introduce a non-terminal $\Sigma_{v,q,q'} \in V'$ which corresponds to a word produced by E, from the non-terminal v, which is read along a path from state q to q' in M. Similarly, for each terminal $x \in \Sigma$, and each $q, q' \in Q$, we introduce a non-terminal $\Sigma_{x,q,q'} \in V'$ which corresponds to a letter, produced by E, which is read along an edge between states q and q' in the string transducer M. To simplify our proof, we allow non-terminals of the form $\Sigma_{x,q,q'}$ where there is no edge from q to q' labelled by a. We only verify that a valid path is being represented at the end of the construction.

We now describe the tables of the grammar E', beginning with an initialisation table as follows.

Initialisation table: τ_{init}.

The table τ_{init} decides on an accepting path in the automaton M and is defined as follows:

$$\tau_{\text{init}}(S') = \{\Sigma_{S,q_0,q'} \mid q' \in A\},$$

and $\tau_{\text{init}}(v) = v$ for all other non-terminals v. Observe that τ_{init} is a table as each of its replacements are finite sets which are all regular languages.

Such a table 'guesses' that the grammar will produce a word which corresponds to a path from q_0 to $q' \in A$. We now describe a modification of the table $\alpha \in T$, and we modify the tables β and γ in precisely the same manner.

Modifying tables $\alpha, \beta, \gamma \in T$.

We introduce a table $\alpha' \in T'$ such that

- for each $v \in V$, and $q, q' \in Q$, we have

$$\Sigma_{s_1,q,q_1}\Sigma_{s_2,q_1,q_2}\Sigma_{s_3,q_2,q_3}\cdots\Sigma_{s_{k+1},q_k,q'} \in \alpha'(\Sigma_{v,q,q'})$$

  for every $q_1, q_2, \ldots, q_k \in Q$, if and only if $s_1 s_2 \cdots s_k \in \alpha(v)$. Note that we only require that the states of adjacent letters match. Thus, it can be seen that the language $\alpha'(\Sigma_{v,q,q'})$ is regular; and
- for all other non-terminals $x \in \Sigma'$, we have $\alpha'(x) = x$.

We perform the same modification to tables $\beta, \gamma \in T$ to obtain tables $\beta', \gamma' \in T'$, respectively.

Observation I:

From the definition of the tables $\tau_{\text{init}}, \alpha', \beta', \gamma' \in T'$, we see that for every choice of states $q, q', q_1, q_2, \ldots, q_k \in Q$, symbols $s_1, s_2, \ldots, s_{k+1} \in \Sigma \cup V$, and $n \in \mathbb{N}$,

$$\Sigma_{s_1,q,q_1}\Sigma_{s_2,q_1,q_2}\Sigma_{s_3,q_2,q_3}\cdots\Sigma_{s_{k+1},q_k,q'} \in S' \cdot \left(\tau_{\text{init}}\alpha'(\beta')^n\gamma'\right)$$

if and only if $q = q_0$, $q' \in A$ and

$$s_1 s_2 s_3 \cdots s_{k+1} \in S \cdot \left(\alpha\beta^n\gamma\right).$$

All that now remains is to introduce one additional table to apply the action of the string transducer M.

Final table: τ_{final}.

The additional table τ_{final} is defined as follows. For each $\Sigma_{a,q,q'}$ with $a \in \Sigma$, we define

$$\tau_{\text{final}}(\Sigma_{a,q,q'}) = \begin{cases} w & q \to^{(a,w)} q' \\ \Sigma_{a,q,q'} & \text{otherwise.} \end{cases}$$

For all other non-terminals $v \in V$, we define $\tau_{\text{finish}}(v) = v$.

Observation II.

For each $w \in \Gamma^*$, we have

$$w \in S' \cdot \left(\tau_{\text{init}} \alpha'(\beta')^n \gamma' \tau_{\text{finish}}\right)$$

if and only if there is some $w' \in \Sigma^*$ such that

$$w' \in \left(\alpha \beta^n \gamma\right)$$

with $q_0 \to^{(w',w)} q$ for some $q \in A$.

Modifying the grammar E'.

We construct the tables $\alpha'' = \tau_{\text{init}}\alpha'$, $\beta'' = \beta'$ and $\gamma'' = \gamma'\tau_{\text{finish}}$. Then, with the rational control $\mathcal{R} = \alpha''(\beta'')^*\gamma''$, we have an ET0L grammar for the language $L' = M(L)$. Moreover, the grammar E' is unambiguous because E is unambiguous, and it is limiting by the construction of the tables. □

We use Proposition 3.8 to prove Proposition 3.11 which is then used in the proof of our main theorem. To prove this proposition, we first require some technical lemmas given as follows. Recall that a subset $W \subset \Gamma^+$ is an *antichain* with respect to prefix order if for each choice of words $u, v \in W$, the word u is not a proper prefix of v.

Lemma 3.9. *Let $W \subset \Gamma^+$ be a finite antichain with respect to prefix order. For each word $w \in W$, we fix a word $x_w \in \Sigma^*$. Define a map $f : \mathcal{P}(\Gamma^*) \to \mathcal{P}(\Sigma^*)$ as*

$$f(L) = \{x_{w_1} x_{w_2} \cdots x_{w_k} \in \Sigma^* \mid w_1 w_2 \cdots w_k \in L \text{ where each } w_i \in W\}.$$

Then, there is a string transducer $M = (\Gamma, \Sigma, Q, A, q_0, \delta)$ such that $f(L) = f(L \cap W^*) = M(L)$. Moreover, such a string transducer is computable from the set W and words $x_w \in \Sigma^*$.

The proof below is standard. In particular, the existence of such an automaton is known since W is a code (cf. [4]), see, for example, [4, pp. 200–202] in which a similar automaton is constructed and called a 'decoding automaton'.

Proof. First we observe that if $W = \emptyset$, then $f(L) = \emptyset$ for each language $L \subseteq \Gamma^*$. In this case, any such string transducer with $A = \emptyset$ satisfies the statement of the lemma. Thus, in the remainder of this proof, we assume that $W \neq \emptyset$.

Let $w \in \Gamma^*$ be a word for which $w \in W^*$. Then, since W is a finite antichain with respect to the prefix order, there is a unique factorisation of w as $w = w_1 w_2 \cdots w_k$ where each $w_i \in W$.

We construct a string transducer $M = (\Gamma, \Sigma, Q, A, q_0, \delta)$ as follows. For each proper prefix $u \in \Gamma^*$ of a word $w \in W$, we introduce a state $q_u \in Q$. The initial state is $q_0 = q_\varepsilon$, and the set of accepting

states is $A = \{q_\varepsilon\}$. Further, our automaton has one additional state q_{fail} which is a fail state; that is,

$$\delta(g, q_{\text{fail}}) = (\varepsilon, q_{\text{fail}})$$

for each $g \in \Gamma$. We then specify the remaining transitions as follows.

For each state q_u with $u \in \Gamma^*$, and each $g \in \Gamma$, we define the transition

$$\delta(g, q_u) = \begin{cases} (x_w, q_\varepsilon) & \text{if } w = ug \in W, \\ (\varepsilon, q_{ug}) & \text{if } ug \text{ is a proper prefix of some } w \in W, \\ (\varepsilon, q_{\text{fail}}) & \text{otherwise.} \end{cases}$$

The string transducer M is now completely specified. It is clear from the construction that $f(L) = M(L)$ for each $L \subseteq \Gamma^*$. Moreover, one observes that every step of this construction is computable. □

Lemma 3.10. *Suppose that G is an infinite group with a finite monoid generating set X. Fix a finite number of words $u_1, u_2, \dots, u_k \in X^*$. Then there exists a choice of non-empty words $w_1, w_2, \dots, w_k \in X^* \setminus \{\varepsilon\}$, such that each $\overline{w_i} = 1$ and*

$$W = \{w_1u_1, w_2u_2, \dots, w_ku_k\}$$

is an antichain in prefix order; that is, for each choice of words $x, y \in W$, the word x is not a proper prefix of y. Moreover, such a choice of set W is computable if the word problem for G is computable.

Proof. We begin by constructing the words $w_1, w_2, \dots, w_k$ as follows.

Let $\alpha_1 \in X$ be a non-trivial generator, that is, $\overline{\alpha_1} \neq 1$; then let $\beta_1 \in X^*$ be a geodesic with $\overline{\alpha_1\beta_1} = 1$ (If X is a symmetric generating set, then we may choose $\beta_1 = \alpha_1^{-1}$.) From this selection, we define $w_1 = \alpha_1\beta_1$. We now choose the words $w_2, w_3, \dots, w_k$ sequentially as follows.

For each $i \geqslant 2$, we choose a geodesic $\alpha_i \in X^*$ with length $|\alpha_i| = |\alpha_{i-1}\beta_{i-1}| + 1$. We then choose a geodesic $\beta_i \in X^*$ such that $\overline{\alpha_i\beta_i} = 1$ (If X is a symmetric generating set, then we may choose $\beta_i = \alpha_i^{-1}$.) Then, define $w_i = \alpha_i\beta_i$.

We have now selected a sequence of words $w_1, w_2, \dots, w_k$. For each word w_i, let γ_i denote the longest prefix which is a geodesic. Then

$$|\gamma_{i-1}| < |w_{i-1}| < |\alpha_i| \leqslant |\gamma_i|$$

for each $i \in \{2, 3, \dots, k\}$. Thus, $|\gamma_1| < |\gamma_2| < \cdots < |\gamma_k|$ and $|\gamma_i| < |w_iu_i|$ for each i.

We now see that, if w_iu_i is a proper prefix of some word $v \in X^*$, then γ_i is also the longest prefix of v which is a geodesic. Hence, we conclude that the set

$$W = \{w_1u_1, w_2u_2, \dots, w_ku_k\}$$

is an antichain as required. All computations are possible as long as the word problem is computable for G. □

Proposition 3.11. *Suppose that the group G has a finite symmetric generating set X, and that $M \subseteq G$ is a subset of the group for which $L = \{w \in X^* \mid \overline{w} \in M\}$ is an unambiguous limiting ET0L language. Then, for each subgroup $H \leqslant G$ with finite generating set $Y \subseteq H$, the set $L' = \{w \in Y^* \mid \overline{w} \in M \cap H\}$ forms an unambiguous limiting ET0L language. Moreover, an unambiguous limiting ET0L grammar for L' is computable from such a grammar for L.*

Proof. We write the set $Y = \{y_1, y_2, \dots, y_m\}$. Then, for each $y_i \in Y$, fix a non-empty word $u_i \in X^*$ for which $\overline{y_i} = \overline{u_i}$. From Lemma 3.10, we then see that there is a choice of words $w_1, w_2, \dots, w_m$, with each $\overline{w_i} = 1$, such that the words w_1u_1, $w_2u_2, \dots, w_mu_m$ form an antichain with respect to prefix order. From our choice of words u_i and w_i, we then see that

$$L' = \left\{ y_{i_1}y_{i_2}\cdots y_{i_k} \in Y^* \middle| (w_{i_1}u_{i_1})(w_{i_2}u_{i_2})\cdots(w_{i_k}u_{i_k}) \in L \right\}.$$

Thus, from Lemma 3.9, we see that there is a string transducer M for which $L' = M(L)$. Moreover, we see that this string transducer is injective. Our result then follows from Proposition 3.8. Moreover, we see that every step of our construction is computable. □

3.3 | Generating functions

A multivariate generating function is a generalisation of the ordinary generating function to several variables. It is used to study sequences indexed by multiple indices, such as $a_{i_1,i_2,\dots,i_k}$. Formally, for a k-dimensional sequence $\{a_{i_1,i_2,\dots,i_k}\}$, the multivariate generating function is defined as

$$g(x_1, x_2, \dots, x_k) = \sum_{i_1=0}^{\infty}\sum_{i_2=0}^{\infty}\cdots\sum_{i_k=0}^{\infty} a_{i_1,i_2,\dots,i_k}\, x_1^{i_1}x_2^{i_2}\cdots x_k^{i_k}.$$

Multivariate generating functions provide a compact way to encode and manipulate multidimensional sequences. They are especially useful in combinatorics, probability, and the study of systems with several interacting parameters (see [22]). The set of such functions is equipped with a natural notion of convergence. Let $\{g_n(x_1, \dots, x_k)\}_{n\geqslant 0}$ be a sequence of multivariate generating functions

$$g_n(x_1, \dots, x_k) = \sum_{i_1,\dots,i_k \geqslant 0} a^{(n)}_{i_1,\dots,i_k} x_1^{i_1}\cdots x_k^{i_k}.$$

Then $g_n \to g$ if, for each $i_1, i_2, \dots, i_k$, we have $a^{(n)}_{i_1,\dots,i_k} \to a_{i_1,\dots,i_k}$.

We are now ready to prove our theorem on the generating functions of ET0L languages as follows.

Theorem 3.12. *Let $L \subseteq \Sigma^*$ be an unambiguous limiting ET0L language. Then, it is computable to find a description of the generating function of L as*

$$f(z) = g(r_1(z), r_2(z), \dots, r_k(z))$$

where each $r_i(z) \in \mathbb{N}[[z]]$ is a rational power series, and $g(x_1, x_2, \dots, x_k)$ is a formal power series defined as

$$g(x_1, x_2, \dots, x_k) = \lim_{n\to\infty} g_n(x_1, x_2, \dots, x_k)$$

where $g_0(x_1, x_2, \dots, x_k) \in \mathbb{N}[[x_1, \dots, x_k]]$ is a rational power series, and

$$g_{n+1}(x_1, x_2, \dots, x_k) = g_n(q_1(x_1, x_2, \dots, x_k), q_2(x_1, x_2, \dots, x_k), \dots, q_k(x_1, x_2, \dots, x_k))$$

for each $n \geqslant 0$ where each $q_i(x_1, \dots, x_k) \in \mathbb{N}[[x_1, \dots, x_k]]$ is a rational series depending on g_n. In the above k is a constant that depends on the grammar.

Remark 3.13. The coefficients of the generating functions g_n correspond to the number of words in the set $S \cdot \alpha\beta^n \in (\Sigma \cup V)^*$. We know that the count of all such words converges from properties 4 and 5 of Definition 3.5. Hence, the limit in the statement of the theorem exists.

Proof. Let $E = (\Sigma, V, T, \mathcal{R}, S)$ be an ET0L language as in Definition 3.5 where $V = \{X_1 = S, X_2, \dots, X_k\}$, $T = \{\alpha, \beta, \gamma\}$ and $\mathcal{R} = \alpha\beta^*\gamma$. Suppose that

$$\alpha : \begin{cases} X_1 \mapsto L_{\alpha,1} \\ X_2 \mapsto L_{\alpha,2} \\ \quad \vdots \\ X_k \mapsto L_{\alpha,k} \end{cases}, \qquad \beta : \begin{cases} X_1 \mapsto L_{\beta,1} \\ X_2 \mapsto L_{\beta,2} \\ \quad \vdots \\ X_k \mapsto L_{\beta,k} \end{cases} \quad \text{and} \quad \gamma : \begin{cases} X_1 \mapsto L_{\gamma,1} \\ X_2 \mapsto L_{\gamma,2} \\ \quad \vdots \\ X_k \mapsto L_{\gamma,k} \end{cases}$$

where each $L_{\alpha,i}$, $L_{\beta,i}$ and $L_{\gamma,i}$ is a regular language over $V \cup \Sigma$.

In the remainder of this proof, we write $\mathbf{x}$ for the tuple of variables $(x_1, x_2, \dots, x_k)$ where each variable x_i corresponds to the variable $X_i \in V$. Suppose that $m = |\Sigma|$, we then write $\mathbf{y}$ for the tuple of variables $(y_1, y_2, \dots, y_m)$ where each variable y_i corresponds to a letter σ_i in $\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_m\}$.

For the regular languages $L_{\alpha,i}$, $L_{\beta,i}$ and $L_{\gamma,i}$ we write $h_{\alpha,i}(\mathbf{x}, \mathbf{y})$, $h_{\beta,i}(\mathbf{x}, \mathbf{y})$ $h_{\gamma,i}(\mathbf{x}, \mathbf{y})$ for their generating functions, respectively. It is well-known that the multivariate generating function of regular languages are rational (this follow from [27, p. 125]).

Then, for each $h_{\gamma,i}$, we define a function $H_{\gamma,i}(z)$ as

$$H_{\gamma,i}(z) = h_{\gamma,i}(\underbrace{0, 0, \dots, 0}_{k \text{ times}}, \underbrace{z, z, \dots, z}_{m \text{ times}}).$$

We then see that the generating function can be written as

$$f(z) = g(H_{\gamma,1}(z), H_{\gamma,2}(z), \dots, H_{\gamma,k}(z), z, z, \dots, z)$$

with

$$g(\mathbf{x}, \mathbf{y}) = \lim_{n\to\infty} g_n(\mathbf{x}, \mathbf{y})$$

where

$$g_0(\mathbf{x},\mathbf{y}) = h_{\alpha,1}(\mathbf{x},\mathbf{y}) \quad \text{and}$$
$$g_{n+1}(\mathbf{x},\mathbf{y}) = g_n(h_{\beta,1}(\mathbf{x},\mathbf{y}), h_{\beta,2}(\mathbf{x},\mathbf{y}), \dots, h_{\beta,k}(\mathbf{x},\mathbf{y}), \mathbf{y})$$

for each $n \geqslant 0$. □

4 | MAIN THEOREM

The content in this section is devoted to proving our main theorem, stated as follows.

Theorem 4.1. *Suppose that we are given a finite symmetric set X of bounded automaton automorphisms acting on the tree $\mathcal{T}_d = C^*$, and words $a, b \in C^*$. Then, we can effectively compute, uniformly over all X, a and b, an ET0L grammar which generates the language $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ with $G = \langle X \rangle$ and $\eta = ab^\omega$, and an ET0L grammar which generates the complement of this language, that is, $X^* \setminus \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$. Moreover, in both cases the grammars are unambiguous limiting.*

We begin by noting that, from Proposition 3.11 and Lemma 2.9, it is sufficient to consider the case of G a bounded automata group with a finite generating set which can be partitioned as $X = F \cup D$ where $F \subset \mathrm{Fin}(\mathcal{T}_d)$ is a set of finitary automorphisms and $D \subset \mathrm{Dir}(\mathcal{T}_d)$ is a set of directed automorphisms. Thus, in the remainder of this section, we will assume so without loss of generality.

In the theorem statement, we are given an eventually periodic ray $\eta = ab^\omega$, specified as two finite-length words $a, b \in C^*$. The goal is to construct two ET0L grammars, $E = (X, V, T, \mathcal{R}, S)$ and $E' = (X, V, T, \mathcal{R}, S')$, for the languages $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ and $X^* \setminus \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$, respectively. Moreover, at the end of our construction, we point out that these grammars are unambiguous limiting. The only difference between the two grammars is their starting symbol.

To describe the non-terminals in our grammar, we need to define a finite set of eventually periodic rays described by a finite set of pairs $I \subset C^* \times C^*$ as follows.

Lemma 4.2. *Let $I \subseteq C^* \times C^*$ be a finite set such that, for each*

$$\zeta \in \{\eta\} \cup \mathrm{spine}(D) \cup (\mathrm{spine}(D) \cdot D)$$

where $\mathrm{spine}(D) = \{\mathrm{spine}(\delta) \mid \delta \in D\}$, there exists some $(u, v) \in I$ such that $\zeta = uv^\omega$. Then it is effectively computable to construct such a set I, where additionally

1. *there exists some $\ell \in \mathbb{N}$ such that $|u| = |v| = \ell$ for each $(u, v) \in I$;*
2. *if $(u, v), (u', v') \in I$ with $(u, v) \neq (u', v')$, then $u \neq u'$;*
3. *for each $x \in D$, there is some $(u, v) \in I$ such that $\mathrm{spine}(x) = uv^\omega$ with $x@u = x@uv$; and*
4. *for each finitary automorphism $f \in F$, we have $\ell \geqslant \mathrm{depth}(f)$.*

Proof. For the moment, if we ignore properties 1–4, it is clear that we may construct such a set I from a description of the generating set of bounded automaton automorphisms X. In particular, given generators $x, y \in X$, it is computable to check if they are not finitary (i.e. there are no

non-trivial cycles among the restrictions in their descriptions), and it is also computable to find the initial and periodic segment of their spines (it follows from the proof of Proposition 2.8). Moreover, the action of y on the spine of x can then be computed using a finite amount of memory.

In the remainder of this proof, we show how to modify such a set so that it satisfies each of the desired properties of the lemma.

Property 1: Let $m \geqslant 0$ be the constant defined as

$$m = \max\{|u| \mid (u, v) \in I\}.$$

Now let $\ell \geqslant 1$ be defined as

$$\ell = \operatorname{lcm}(\{|v| \mid (u, v) \in I\} \cup \{m\}).$$

We now construct a finite set I' as follows. For each $(u, v) \in I$, we define u' as the length-ℓ prefix of uv^ω, and we define v' to be a cyclic permutation of the word v for which $u'(v')^\omega = uv^\omega$. We add $(u', (v')^{\ell/|v'|})$ to the set I'.

We now see that if $(a, b) \in I'$, then $|a| = |b| = \ell$ and that I' represents the same elements as I. We also see that I' is minimal since for each $(a, b), (a', b') \in I'$, we have $ab^\omega = a'(b')^\omega$ if and only if $(a, b) = (a', b')$. The above steps are computable. Thus, given a set I, there is an algorithm that can construct the set I'.

We replace I with I', then in the remainder of this proof, we assume that there is some $\ell \geqslant 1$ such that for each $(u, v) \in I$, we have $|u| = |v| = \ell$.

Property 2: For each $(u, v), (u', v') \in I$ with $(u, v) \neq (u', v')$, we have $uv \neq u'v'$. We construct a set $I' \subset C^* \times C^*$ as

$$I' = \{(uv, vv) \mid (u, v) \in I\}.$$

We note that the set I' has properties 1 and 2. Moreover, given a set I, there is an algorithm which can construct the set I'. Thus, we replace I with I', and ℓ with 2ℓ we may assume that our set I satisfies properties 1 and 2.

Properties 3: By Proposition 2.8, for each $x \in D$, we can choose words $u_x, v_x \in C^*$ such that $\operatorname{spine}(x) = u_x v_x^\omega$ with $x@u_x = x@u_x v_x$.

Properties 4: We define a constant $\ell' \geqslant 1$ as

$$\ell' = \operatorname{lcm}(\{\ell\} \cup \{|u_x| \mid x \in D\} \cup \{|v_x| \mid x \in D\} \cup \{\operatorname{depth}(f) \mid f \in F\}).$$

We now define a set I' as

$$I' = \{(uv^{(\ell'/\ell)-1}, v^{\ell'/\ell}) \mid (u, v) \in I\}.$$

We then see that this new set I' satisfies the properties 1 and 2, and also satisfies property 3.

Observe that the above steps are computable, that is, given a set I and a description of the generators D, there is an algorithm which can construct the set I'. Thus, after replacing I with I' and ℓ with ℓ', we may assume that I satisfies all of our desired properties. □

Our grammars have three tables $T = \{\tau_{\text{init}}, \tau_{\text{up}}, \tau_{\text{finish}}\}$ and rational control $\mathcal{R} = \tau_{\text{init}}(\tau_{\text{up}})^* \tau_{\text{finish}}$.

The construction of the tables will be based on directional depth as defined in Definition 2.5. To proceed with it, we introduce the following tool.

Definition 4.3. Let $w = s_1 s_2 \cdots s_k \in X^*$, where each $s_i \in X$, and let $\zeta \in C^* \cup C^\omega$.

Define the word w *decorated with respect to* ζ as

$$\mathrm{Dec}(\zeta, w) := s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)} \quad \text{where} \quad a_i = \mathrm{DDepth}(\zeta_i, s_i)$$

with $\zeta_1 = \zeta$ and $\zeta_{i+1} = \zeta_i \cdot s_i$ for each $i \in \{1, 2, \dots, k\}$.

Before proceeding with the construction, let us give an overview of how it is intended to work. Suppose $w = s_1 s_2 \cdots s_k$ is a word generated by the grammar E with

$$\mathrm{Dec}(\zeta, w) := s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}. \tag{2}$$

Then, when producing w, our grammar fills in the letters s_i in decreasing order of a_i. In particular, let $\ell \in \mathbb{N}$ be the constant derived in Lemma 4.2, and suppose that $A \in \mathbb{N}$ is chosen such that $\lceil a_i/\ell \rceil \leqslant A + 3$ for each finite $a_i < \infty$. Then, our grammar will produce the word w as

$$w \in S \cdot \tau_{\mathrm{init}} \left(\tau_{\mathrm{up}}\right)^A \tau_{\mathrm{final}}.$$

In particular: *(note that in the following a_i are as in (2))*

- The letters s_i with $a_i = \infty$ are generated when the table τ_{init} is applied.
- The letters s_i with $3\ell < a_i < \infty$ first come into a sentential form after applying the last τ_{up} of

$$\tau_{\mathrm{init}} \left(\tau_{\mathrm{up}}\right)^{A+4-\lceil a_i/\ell \rceil}, \tag{3}$$

  that is, for each m, letters s_i with $(3+m)\ell < a_i \leqslant (4+m)\ell$ enter a sentential form after applying

$$\tau_{\mathrm{init}} \left(\tau_{\mathrm{up}}\right)^{A-m}.$$

  In (3) we have $A + 4$ in the exponent as in this case we have $4 \leqslant \lceil a_i/\ell \rceil \leqslant A + 3$, and thus it would follow that $1 \leqslant A + 4 - \lceil a_i/\ell \rceil \leqslant A$, that is, the sequence of tables contains at least one τ_{up}.
- The letters s_i with $a_i \leqslant 3\ell$ are generated at the end of the production by the table τ_{final}.

Our construction of the grammar E' satisfies analogous properties. During each part of this production, we ensure that words are produced unambiguously with respect to their rational control.

4.1 | Non-terminals and starting symbols

We begin by introducing the starting symbols $S = [\![\eta; \eta]\!]$ and $S' = [\![\eta; \neg\eta]\!]$ of the grammars E and E', respectively. The non-terminal S is a placeholder for a word whose action stabilises the ray η.

Similarly, the non-terminal S' is a placeholder for a word whose action does not stabilise η, that is, whose action takes η to a different ray. We now define the remaining non-terminals as follows.

In the following, it should be understood that I refers to a set of pairs as constructed in Lemma 4.2.

4.1.1 | Non-terminals with restrictions on result of a word action

Each of the non-terminals which we introduce below can be a placeholder for words which have very particular actions on a particular vertex of the tree $\mathcal{T}_d$. The main idea of our construction of the grammar E is to decompose the action of a word on η into a sequence of actions of this form.

For each pair of rays $\zeta = uv^\omega, \zeta' = u'(v')^\omega$ where $(u, v), (u', v') \in I$ and each pair of paths $p, p' \in C^{2\ell}$, we have non-terminals $[\![\zeta, p; \zeta', p']\!]$ and $[\![\zeta, p; \neg\eta]\!]$.

In our proof, we construct our grammars so that for each $m \geqslant 0$, we have

$$w \in X^* \cap ([\![\zeta, p; \zeta', p']\!] \cdot (\tau_{\text{up}})^m \tau_{\text{finish}})$$

if and only if $uv^m p \cdot \overline{w} = u'(v')^m p'$ and

$$\text{Dec}(uv^m p, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}$$

where each $a_i \leqslant |uv^m p| = (m+3)\ell$. Moreover, our grammar ensures that, for each $m \geqslant 0$, we have

$$w \in X^* \cap ([\![\zeta, p; \neg\eta]\!] \cdot (\tau_{\text{up}})^m \tau_{\text{finish}})$$

if and only if $uv^m p \cdot \overline{w} \neq ab^{m+2}$ with $(a, b) \in I$ and $ab^\omega = \eta$, and that

$$\text{Dec}(uv^m p, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}$$

where each $a_i \leqslant |uv^m p| = (m+3)\ell$. That is, $[\![\zeta, p; \neg\zeta]\!]$ corresponds to words whose action does not take words of the form $uv^m p$ to a prefix of η.

Let $ab^\omega = \eta$ where $(a, b) \in I$, then we see that the non-terminals $[\![\eta, b^2; \eta, b^2]\!]$ and $[\![\eta, b^2; \neg\eta]\!]$ can be used to generate subwords of words in $\mathsf{WP}(G, X, \text{Stab}(\eta))$ and in $X^* \setminus \mathsf{WP}(G, X, \text{Stab}(\eta))$, respectively, which are composed of letters s_j with $a_j < \infty$ as in (4.3)

4.1.2 | Additional non-terminals for E'

For technical reasons, for the grammar E', we require some additional non-terminals which are placeholders for factors of output words containing letters s_j with each $a_j < \infty$ as in (4.3).

For each ray $\zeta = uv^\omega$ with $(u, v) \in I$ and each path $p \in C^{2\ell}$, we introduce a non-terminal $[\![\zeta, p; \text{any}]\!]$. We construct our grammar such that for each $m \geqslant 0$, we have

$$w \in X^* \cap ([\![\zeta, p; \text{any}]\!] \cdot (\tau_{\text{up}})^m \tau_{\text{finish}})$$

if and only if

$$\mathrm{Dec}(uv^m p, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}$$

where each $a_i \leqslant |uv^m p| = (m+3)\ell$.

4.1.3 | Intuition of the starting symbols

In our proof, we construct the tables of our grammar in such a way that, for each $m \geqslant 0$, we have

$$w \in X^* \cap (\llbracket \eta; \eta \rrbracket \cdot \tau_{\mathrm{init}} (\tau_{\mathrm{up}})^m \tau_{\mathrm{finish}})$$

if and only if $w \in \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ and

$$\mathrm{Dec}(\eta, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}$$

where each $a_i \in \{0, 1, 2, 3, \ldots, (m+3)\ell\} \cup \{\infty\}$. For each $m \geqslant 0$,

$$w \in X^* \cap (\llbracket \eta; \neg\eta \rrbracket \cdot \tau_{\mathrm{init}} (\tau_{\mathrm{up}})^m \tau_{\mathrm{finish}})$$

if and only if $w \in X^* \setminus \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ and

$$\mathrm{Dec}(\eta, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}$$

where each $a_i \in \{0, 1, 2, 3, \ldots, (m+3)\ell\} \cup \{\infty\}$. From these properties, it is clear that the grammars E and E' satisfy the limiting property ((5)) as in Definition 3.5. Moreover, from our choice of tables and rational control, we see that these grammars satisfy properties ((1)) and ((2)) in Definition 3.5. In the remainder of our construction, we verify that it indeed satisfies the remaining properties of Definition 3.5.

4.1.4 | Computability of non-terminals

Given the set $I \subset C^* \times C^*$ and a description of the ray $\eta = ab^\omega$, we can list all of the finite non-terminals.

4.2 | Initialisation map: τ_{init}

To explain the constructions of the initialisation tables of E and E', we begin with the following observation. Let $w = s_1 s_2 \cdots s_k \in X^*$ and consider its decorated version with respect to η

$$\mathrm{Dec}(\eta, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)} \tag{4}$$

with $a_i \in \mathbb{N} \cup \{\infty\}$. Then factor the word w uniquely as

$$w = z_0 x_1 z_1 x_2 z_2 \cdots x_q z_q \tag{5}$$

where each $z_j \in X^*$, each $x_j \in D$ where the words z_j correspond to (potentially empty) sequences of letters s_i for which $a_i < \infty$, and each x_j correspond to letters s_i for which $a_i = \infty$. We define our initialisation table such that it fills in each x_j with a directed automorphism, and puts an appropriate placeholder non-terminal in the spot of each word z_j.

4.2.1 | Initialising the grammar E

Let $\eta = ab^\omega$ where $(a, b) \in I$, then from the factorisation in (5) we define the language $[\![\eta, \eta]\!] \cdot \tau_{\text{init}}$ to contain all words of the form

$$[\![\eta, b^2; \alpha_1, (\upsilon_1)^2]\!]\, x_1\, [\![\alpha_1', (\upsilon_1')^2; \alpha_2, (\upsilon_2)^2]\!]\, x_2\, [\![\alpha_2', (\upsilon_2')^2; \alpha_3, (\upsilon_3)^2]\!]$$
$$x_3\, [\![\alpha_3', (\upsilon_3')^2; \alpha_4, (\upsilon_4)^2]\!] \cdots x_k\, [\![\alpha_k', (\upsilon_k')^2; \eta, b^2]\!] \in ([\![\eta, \eta]\!] \cdot \tau_{\text{init}})$$

where

(1) $\text{spine}(x_i) = \alpha_i = u_i(\upsilon_i)^\omega$ where $(u_i, \upsilon_i) \in I$ for each $i \in \{1, 2, \dots, k\}$;
(2) $\text{spine}(x_i) \cdot x_i = \alpha_i' = u_i'(\upsilon_i')^\omega$ where $(u_i', \upsilon_i') \in I$ for each $i \in \{1, 2, \dots, k\}$; and
(3) $b = \upsilon_1 = \upsilon_k'$ and $\upsilon_i' = \upsilon_{i+1}$ for each $i \in \{1, 2, \dots, k-1\}$.

Each non-terminal of the form $[\![-, -; -, -]\!]$ as above corresponds to some word z_i as in (5).

Item (3) ensures that the action of the letters x_i are being tracked correctly, in particular, the words corresponding to the placeholders $[\![-, -; -, -]\!]$ cannot modify the ray beyond a particular finite depth: Item (3) ensures that the tail of these rays match.

The set of words $[\![\eta, \eta]\!] \cdot \tau_{\text{init}} \subseteq (X \cup V)^*$, as defined above, is a regular language since, for each word in the language, the possible values of each letter depends, at most, on the previous two non-terminals. Thus, one could construct a finite-state automaton to recognise all such words where the states of the automaton correspond to the possible values of the two previous non-terminals of the form $[\![-, -; -, -]\!]$.

4.2.2 | Initialising the grammar E'

Let $\eta = ab^\omega$ where $(a, b) \in I$, then from the factorisation in (5) we define the language $[\![\eta, \neg\eta]\!] \cdot \tau_{\text{init}}$ to contain all words of the form

$$[\![\eta, b^2; \alpha_1, (\upsilon_1)^2]\!]\, x_1\, [\![\alpha_1', (\upsilon_1')^2; \alpha_2, (\upsilon_2)^2]\!]\, x_2\, [\![\alpha_2', (\upsilon_2')^2; \alpha_3, (\upsilon_3)^2]\!]$$
$$x_3\, [\![\alpha_3', (\upsilon_3')^2; \alpha_4, (\upsilon_4)^2]\!] \cdots x_k\, [\![\alpha_k', (\upsilon_k')^2; \underline{\varphi}]\!] \in ([\![\eta, \neg\eta]\!] \cdot \tau_{\text{init}})$$

where

(1) $\text{spine}(x_i) = \alpha_i = u_i(\upsilon_i)^\omega$ where $(u_i, \upsilon_i) \in I$ for each $i \in \{1, 2, \dots, k\}$;

(2) $\text{spine}(x_i) \cdot x_i = \alpha'_i = u'_i(v'_i)^\omega$ where $(u'_i, v'_i) \in I$ for each $i \in \{1, 2, \dots, k\}$;
(3) $b = v_2$ and $v'_i = v_{i+1}$ for each $i \in \{1, 2, \dots, k-1\}$; and
(4) if $v'_k = b$, then $\underline{\varphi} = \neg\eta$, otherwise, $v'_k \neq b$ and $\underline{\varphi} = \text{any}$.

Each non-terminal in the above corresponds to some factor z_i as in (5). Items (1) and (2) ensure that it is possible for each x_i to have infinite directional depth; item (3) and (4) ensure that the word has an action which does not stabilise the ray η, and that this action is being correctly tracked.

The set $[\![\eta, \neg\eta]\!] \cdot \tau_{\text{init}} \subseteq (X \cup V)^*$ is regular for precisely the same reasons as $[\![\eta, \eta]\!] \cdot \tau_{\text{init}}$ is regular in Section 4.2.1. That is, each possible choice of non-terminal depends, at most, on the previous two non-terminals of the form $[\![-, -; -, -]\!]$.

4.2.3 | Validity and computability

From our definition of τ_{init}, we see that both $[\![\eta, \eta]\!] \cdot \tau_{\text{init}}$ and $[\![\eta, \neg\eta]\!] \cdot \tau_{\text{init}}$ are regular languages (cf. Sections 4.2.1 and 4.2.2), and thus τ_{init} is a table as in Definition 3.1. For all other non-terminals, the table τ_{init} does not need to be specified, as this table will only be applied to the starting symbols.

Given a set $I \subset C^* \times C^*$ and a set of non-terminals, there is an algorithm which can generate the finite state automata for the table τ_{init}.

4.3 | Processing map: τ_{up}

We now describe the map τ_{up} that performs replacements on the non-terminals of the form $[\![\zeta, p; \zeta', p']\!]$, $[\![\zeta, p; \neg\eta]\!]$ and $[\![\zeta, p; \text{any}]\!]$. Let $\zeta = uv^\omega$ where $(u, v) \in I$. To simplify the explanation of this map, we begin by giving a sketch of the intended meaning of each such non-terminal.

Recall from Sections 4.1.1 and 4.1.2 that if the word $w \in X^*$ belongs to the set

$$X^* \cap ([\![\zeta, p; \zeta', p']\!] \cdot (\tau_{\text{up}})^m \tau_{\text{finish}}),\ X^* \cap ([\![\zeta, p; \neg\eta]\!] \cdot (\tau_{\text{up}})^m \tau_{\text{finish}}) \text{ or}$$
$$X^* \cap ([\![\zeta, p; \text{any}]\!] \cdot (\tau_{\text{up}})^m \tau_{\text{finish}}),$$

for some $m \geqslant 0$, then we can decorate the word w as

$$\text{Dec}(uv^m p, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)}$$

where each $a_i \leqslant |uv^m p| = (m+3)\ell$. Analogously to (5) in Section 4.2, w can be uniquely factored as

$$w = z_0 x_1 z_1 x_2 z_2 \cdots x_s z_s \tag{6}$$

where each $z_j \in X^*$ contains the letters s_i of the word w for which $a_i \leqslant (m+2)\ell$, and each x_j is a letter s_i of the word w for which a_i is bounded as $(m+2)\ell < a_i \leqslant (m+3)\ell$. From Lemma 4.2, we see that each x_j must be a directed automorphism in $D = X \cap \text{Dir}(\mathcal{T}_d)$ as the corresponding letter s_i has $a_i > \ell$.

We define the map τ_{up} so that it interprets the non-terminal $[\![\zeta, p; \zeta', p']\!]$, $[\![\zeta, p; \neg\eta]\!]$ and $[\![\zeta, p; \text{any}]\!]$ as words of the form (6), by producing words where each z_i is represented by some

placeholder of the form $[\![\zeta, p; \zeta', p']\!]$, $[\![\zeta, p; \neg\eta]\!]$ or $[\![\zeta, p; \text{any}]\!]$, and each x_i is replaced by an appropriate member of $D = X \cap \text{Dir}(\mathcal{T}_d)$. We now construct the table τ_{up} as follows.

4.3.1 | Case 1: $[\![\zeta, p; \zeta', p']\!]$

Let $\zeta = uv^\omega, \zeta' = u'(v')^\omega$ where $(u, v), (u', v') \in I$, and let $p, p' \in C^{2\ell}$ be paths in the tree. We then define $[\![\zeta, p; \zeta', p']\!] \cdot \tau_{\text{up}}$ such that it contains all non-empty words of the form

$$[\![\zeta, vp_1; \alpha_1, q_1]\!]\, x_1\, [\![\alpha_1', q_1'; \alpha_2, q_2]\!]\, x_2\, [\![\alpha_2', q_2'; \alpha_3, q_3]\!] \cdots x_k\, [\![\alpha_k', q_k'; \zeta', v'p_1']\!] \in ([\![\zeta, p; \zeta', p']\!] \cdot \tau_{\text{up}})$$

where

(1) $vp_1 \in C^{2\ell}$ is the length-2ℓ prefix of $vp \in C^{3\ell}$;
(2) $v'p_1' \in C^{2\ell}$ is the length-2ℓ prefix of $v'p' \in C^{3\ell}$;
(3) each $\alpha_i = \text{spine}(x_i) = u_i(v_i)^\omega$ where $(u_i, v_i) \in I$ and $q_i \in C^{2\ell}$;
(4) each $\alpha_i' = \text{spine}(x_i) \cdot x_i = u_i'(v_i')^\omega$ where $(u_i', v_i') \in I$ and $q_i' \in C^{2\ell}$; and
(5) there is a sequence of words $y_0, y_1, \ldots, y_k \in C^\ell$ such that
   - y_0 is the length-ℓ suffix of $p \in C^{2\ell}$,
   - y_k is the length-ℓ suffix of $p' \in C^{2\ell}$,

   and

$$(u_i v_i q_i y_{i-1}) \cdot x_i = u_i' v_i' q_i' y_i$$

for each $i \in \{1, 2, \ldots, k\}$ such that

$$4\ell = |u_i v_i q_i| < \text{DDepth}(u_i v_i q_i y_{i-1}, x_i) \leqslant |u_i v_i q_i y_{i-1}| = 5\ell \tag{7}$$

for each $i \in \{1, 2, \ldots, k\}$.

Items ((3), (4), (5)) above imply that for each $m \geqslant 0$, we have

$$(u_i (v_i)^m q_i y_{i-1}) \cdot x_i = u_i' (v_i')^m q_i' y_i$$

for each $i \in \{1, 2, \ldots, k\}$ such that

$$(m+3)\ell = |u_i (v_i)^m q_i| < \text{DDepth}(u_i (v_i)^m q_i y_{i-1}, x_i) \leqslant (m+4)\ell$$

for each $i \in \{1, 2, \ldots, k\}$.

The set of words $[\![\zeta, p; \zeta', p']\!] \cdot \tau_{\text{up}}$ forms a regular language as the possible values of each letter depends, at most, on the previous two letters and the previous word of the form y_i as in Item (5) as above. Thus, we may construct a finite-state automaton to recognise all such words.

The words of this regular language, as described above, exactly correspond to words of the form (6). In particular, the non-terminals correspond to the words z_j in and the letters x_j correspond to the letters x_j in (6). Our restrictions ensure that the letters x_j have the appropriate directional depth in (7), and that the action of the associated words matches the action intended by the placeholder $[\![\zeta, p; \zeta', p']\!]$.

4.3.2 | Case 2: $[\![\zeta, p; \neg\eta]\!]$

Let $\zeta = uv^\omega$ where $(u, v) \in I$, and let $p \in C^{2\ell}$ be a path in the tree. We then define $[\![\zeta, p; \neg\eta]\!] \cdot \tau_{\text{up}}$ such that it contains all words of the form

$$[\![\zeta, vp_1; \alpha_1, q_1]\!]\, x_1\, [\![\alpha_1', q_1'; \alpha_2, q_2]\!]\, x_2\, [\![\alpha_2', q_2'; \alpha_3, q_3]\!]\, x_3\, [\![\alpha_3', q_3'; \alpha_4, q_4]\!] \cdots x_k\, [\![\alpha_k', q_k'; \underline{\varphi}]\!]$$

$$\in ([\![\zeta, p; \neg\eta]\!] \cdot \tau_{\text{up}})$$

where

(1) $vp_1 \in C^{2\ell}$ is the length-2ℓ prefix of $vp \in C^{3\ell}$;
(2) each $\alpha_i = \text{spine}(x_i) = u_i(v_i)^\omega$ where $(u_i, v_i) \in I$, and $q_i \in C^{2\ell}$;
(3) each $\alpha_i' = \text{spine}(x_i) \cdot x_i = u_i'(v_i')^\omega$ where $(u_i', v_i') \in I$, and $q_i' \in C^{2\ell}$;
(4) there is a sequence of words $y_0, y_1, \dots, y_k \in C^\ell$ defined such that
   - y_0 is the length-ℓ suffix of $p \in C^{2\ell}$,

   and

$$(u_i v_i q_i y_{i-1}) \cdot x_i = u_i' v_i' q_i' y_i$$

   for each $i \in \{1, 2, \dots, k\}$ such that

$$4\ell = |u_i v_i q_i| < \text{DDepth}(u_i v_i q_i y_{i-1}, x_i) \leqslant |u_i v_i q_i y_{i-1}| = 5\ell$$

   for each $i \in \{1, 2, \dots, k\}$; and
(5) the value of $\underline{\varphi}$ depends on the value of y_k, as in item (4), in particular,

$$\underline{\varphi} = \begin{cases} \neg\eta & \text{if } y_k = b \text{ where } \eta = ab^\omega \text{ with } (a, b) \in I \\ \text{any} & \text{otherwise.} \end{cases}$$

Items 4 and 5 above ensure that the action of the word does not stabilise the ray η, and that each letter x_i has a directional depth within ℓ of the maximum. In particular, items (2, 3, 4) above imply that, for each $m \geqslant 0$,

$$(u_i(v_i)^m q_i y_{i-1}) \cdot x_i = u_i'(v_i')^m q_i' y_i$$

for each $i \in \{1, 2, \dots, k\}$ such that

$$(m+3)\ell = |u_i(v_i)^m q_i| < \text{DDepth}(u_i(v_i)^m q_i y_{i-1}, x_i) \leqslant (m+4)\ell$$

for each $i \in \{1, 2, \dots, k\}$.

Using the same argument as in Section 4.3.1, we see that the set of words, described above, is a regular language. In particular, the possible values of each letter can depend on, at most, the previous letters, and on the previous choice of word y_i as in Item 4 as above. Thus, we can construct a finite-state automaton to recognise all such words.

4.3.3 | Case 3: $[\![\zeta, p; \text{any}]\!]$

Let $\zeta = uv^\omega$ where $(u, v) \in I$, and let $p \in C^{2\ell}$ be a path in the tree. We then define $[\![\zeta, p; \text{any}]\!] \cdot \tau_{\text{up}}$ such that it contains all words of the form

$$[\![\zeta, vp_1; \alpha_1, q_1]\!]\, x_1\, [\![\alpha'_1, q'_1; \alpha_2, q_2]\!]\, x_2\, [\![\alpha'_2, q'_2; \alpha_3, q_3]\!] \cdots x_k\, [\![\alpha'_k, q'_k; \text{any}]\!] \in ([\![\zeta, p; \text{any}]\!] \cdot \tau_{\text{up}})$$

where

(1) $vp_1 \in C^{2\ell}$ is the length-2ℓ prefix of $vp \in C^{3\ell}$;
(2) each with $\alpha_i = \text{spine}(x_i) = u_i(v_i)^\omega$ where $(u_i, v_i) \in I$, and $q_i \in C^{2\ell}$;
(3) each with $\alpha'_i = \text{spine}(x_i) \cdot x_i = u'_i(v'_i)^\omega$ where $(u'_i, v'_i) \in I$, and $q'_i \in C^{2\ell}$; and
(4) there is a sequence of paths $y_0, y_1, \ldots, y_k \in C^\ell$ defined such that
   - y_0 is the length-ℓ suffix of $p \in C^{2\ell}$,

   and

$$(u_i v_i q_i y_{i-1}) \cdot x_i = u'_i v'_i q'_i y_i$$

   for each $i \in \{1, 2, \ldots, k\}$ such that

$$4\ell = |u_i v_i q_i| < \text{DDepth}(u_i v_i q_i y_{i-1}, x_i) \leqslant |u_i v_i q_i y_{i-1}| = 5\ell$$

   for each $i \in \{1, 2, \ldots, k\}$.

The items ((2), (3), (4)) imply that for each $m \geqslant 0$, we have

$$(u_i (v_i)^m q_i y_{i-1}) \cdot x_i = u'_i (v'_i)^m q'_i y_i$$

for each $i \in \{1, 2, \ldots, k\}$ such that

$$(m+3)\ell = |u_i (v_i)^m q_i| < \text{DDepth}(u_i (v_i)^m q_i y_{i-1}, x_i) \leqslant (m+4)\ell$$

for each $i \in \{1, 2, \ldots, k\}$.

Using the same argument as in Sections 4.3.1 and 4.3.2, we see that the set of words, described above, is a regular language. In particular, the possible values of each letter can depend on, at most, the previous letters, and on the previous choice of word y_i as in Item (4) as above. Thus, we can construct a finite-state automaton to recognise all such words.

4.3.4 | Computability

Given the set $I \subset C^* \times C^*$ and the set of non-terminals, there is an algorithm that generates the finite state automata described in Sections 3.1–3.3 of this proof.

4.4 | Final map: τ_{finish}

We now complete the description of our tables by constructing the table τ_{finish}. After applying a sequence of tables of the form $\tau_{\text{init}}(\tau_{\text{up}})^*$, any word will contain at least one non-terminal which is a placeholder for words with a particular action. This table attempts to finish the production of

words which correspond to factors of w containing letters s_j for which $a_j \leqslant 3\ell$ as in (4). We note then that if it is not possible to fill in a non-terminal, then the table leaves it unchanged, and thus does not produce a word as output.

Let $(a, b) \in I$ be such that $\eta = ab^\omega$. For each $\zeta = uv^\omega, \zeta' = u'(v')^\omega$ where $(u, v), (u', v') \in I$ and all paths $p, p' \in C^{2\ell}$, we define the table τ_{finish} as

$$(\llbracket \zeta, p; \zeta', p' \rrbracket \cdot \tau_{\text{finish}}) = \left\{ w \in X^* \;\middle|\; \begin{array}{r} (up) \cdot w = u'p' \text{ and} \\ \text{Dec}(up, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)} \\ \text{where each } a_i \leqslant |up| \end{array} \right\} \cup \{\llbracket \zeta, p; \zeta', p' \rrbracket\},$$

$$(\llbracket \zeta, p; \neg\eta \rrbracket \cdot \tau_{\text{finish}}) = \left\{ w \in X^* \;\middle|\; \begin{array}{r} (up) \cdot w \neq a(b)^2 \text{ and} \\ \text{Dec}(up, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)} \\ \text{where each } a_i \leqslant |up| \end{array} \right\} \cup \{\llbracket \zeta, p; \neg\eta \rrbracket\},$$

$$(\llbracket \zeta, p; \text{any} \rrbracket \cdot \tau_{\text{finish}}) = \left\{ w \in X^* \;\middle|\; \begin{array}{r} \text{Dec}(up, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)} \\ \text{where each } a_i \leqslant |up| \end{array} \right\} \cup \{\llbracket \zeta, p; \text{any} \rrbracket\}.$$

Each of the above is a regular language, in particular, can be recognised by a finite-state automaton with states of the form $q \in C^{3\ell}$, and thus, these automata have at most $d^{3\ell}$ states where $3\ell = |up|$.

In the above definitions, we allow each map to potentially leave a non-terminal unchanged. We add this possibility so that we satisfy property (4) of Definition 3.5.

4.5 | Proof of main theorem

We see from our construction that

$$w \in X^* \cap (S \cdot \tau_{\text{init}}(\tau_{\text{up}})^m \tau_{\text{finish}})$$

for some $m \in \mathbb{N}$ if and only if both $w \in \text{WP}(G, X, \text{Stab}(\eta))$ and

$$\text{Dec}(\eta, w) = s_1^{(a_1)} s_2^{(a_2)} \cdots s_k^{(a_k)} \tag{8}$$

where each $a_i \in \{0, 1, 2, \ldots, (m+3)\ell\} \cup \{\infty\}$. For each $w \in \text{WP}(G, X, \text{Stab}(\eta))$, there exists some m, as in (8). With each application of the tables τ_{init}, τ_{up}, τ_{finish}, we uniquely factor the word w into finitely many subwords. That is, each word generated by our grammar is generated unambiguously with respect to the rational control.

From Section 4.1.3 we see that our grammars satisfy properties (1), (2) and (5) from Definition 3.5. Moreover, from Section 4.4 we see that the grammars also satisfy property (4) from Definition 3.5. Further, from the description of the table τ_{up}, we see that they also satisfy property (3) of Definition 3.5.

The non-terminals and tables of the grammars are computable. Thus, we conclude that the grammars E and E' for the languages $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ and $X^* \setminus \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$, respectively, are unambiguous limiting ET0L, and are effectively computable.

5 | ARE STABILISERS OF INFINITE RAYS CONTEXT-FREE?

Since context-free languages are ET0L, it is natural to ask if Theorem 4.1 can be sharpened to context-free rather than ET0L. Given a bounded automata group $G \leqslant \mathrm{Aut}(\mathcal{T}_d)$ with a finite generating set X and η any infinite ray in $\mathcal{T}_d$, we provide two obstructions to the language $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ being context-free. Exploiting results and techniques from [7], for a large class of groups, we show that almost all of these languages are not context-free. For a few key examples, we show that all such languages are not context-free. Hence, Theorem 4.1 cannot be improved from ET0L to context-free languages.

Recall that a group $G \leqslant \mathrm{Aut}(\mathcal{T}_d)$ has an induced action on the boundary of the rooted regular tree. We can thus consider the family of Schreier graphs associated with this action, as in Section 1. Recall also that $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is the language of words that label closed paths from η to η in the (rooted) Schreier graph Γ_η of $\mathrm{Stab}(\eta)$.

Definition 5.1. Let Γ_{v_0} be a labelled graph rooted in v_0 and let v be a vertex of Γ_{v_0}. The *end-cone* $\Gamma_{v_0}(v)$ is the connected component of $\Gamma_{v_0} \setminus B_{\Gamma_{v_0}}(v_0, |v|)$ which contains v, where $B_{\Gamma_{v_0}}(v_0, k)$ denotes the (open) ball of radius k centred at v_0 and $|v|$ is the distance from v_0 to v. We denote by $\Delta_{v_0}(v)$ the set of vertices of $\Gamma_{v_0}(v)$ that are at a minimal distance from v_0, and we call them *frontier points* of the end-cone.

We say that two end-cones $\Gamma_{v_0}(v_1)$ and $\Gamma_{v_0}(v_2)$ *have the same type* if there exists a graph isomorphism $\varphi : \Gamma_{v_0}(v_1) \to \Gamma_{v_0}(v_2)$ between them that respects the labelling and for which $\varphi(\Delta_{v_0}(v_1)) = \Delta_{v_0}(v_2)$.

Definition 5.2. A rooted graph is context-free if it has finitely many types of end-cones.

On one hand, if $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is context-free, then so is the corresponding Schreier graph. On the other hand, context-free graphs are quasi-isometric to trees (see [23, Propositions 7 and 9]). Hence, we have our first criterion, as follows.

Theorem 5.3 see [23]. *Let G be a finitely generated bounded automata group and let η be an infinite ray in $\partial\mathcal{T}_d$. If $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is context-free, then the Schreier graph Γ_η is quasi-isometric to a tree.*

We will now look at the number of ends, which is known to be a quasi-isometric invariant. It is proven in [7, Corollary 5] that Schreier graphs $(G, X, \mathrm{Stab}(\eta))$ of a bounded automata self-similar group G have either almost surely one end or almost surely two ends. Almost surely here means for almost all $\eta \in \partial\mathcal{T}_d$ with respect to the uniform measure on $\partial\mathcal{T}_d$. Let us first discuss the case of one end. It is straightforward that a one-ended tree is quasi-isometric to either the half line or the half line with infinitely many finite paths of unbounded lengths attached. For the first option, we will prove that a self-similar bounded automata group which is level-transitive cannot have more than two Schreier graphs quasi-isometric to a half-line. For the second option, it is not hard to see that the graph does not have finitely many end-cone types.

In what follows, we will use results and methods from [6, 7]. In particular, we require our automata groups to be self-similar.

It will be helpful to us to describe the Schreier graph Γ_ξ, with $\xi = a_1 a_2 \cdots$, $a_i \in C$, in terms of the finite Schreier graphs $(G, X, \mathrm{Stab}(a_1 a_2 \cdots a_n))$ that correspond to the stabilisers of the vertices of the tree $\mathcal{T}_d$ that lie on the infinite ray ξ. If the action of G on $\mathcal{T}_d$ is transitive on every level C^n, then the Schreier graphs associated to $(G, X, \mathrm{Stab}(a_1 a_2 \cdots a_n))$ and $(G, X, \mathrm{Stab}(\widetilde{a}_1 \widetilde{a}_2 \cdots \widetilde{a}_n))$ are isomorphic as unrooted graphs; we will therefore denote such a graph simply by Γ^n. The vertices of Γ^n are exactly the vertices of the nth level, C^n, and two vertices $a_1 a_2 \cdots a_n$ and $\widetilde{a}_1 \widetilde{a}_2 \cdots \widetilde{a}_n$ are joined by an edge whenever there exists an element of X sending one to the other. By Equation ($\diamond$) in the introduction, the sequence of rooted graphs $\{(\Gamma^n, a_1 a_2 \cdots a_n)\}_n$ converges to the rooted graph (Γ_ξ, ξ) in local topology. This means that for every radius r, the ball $B_{\Gamma_\xi}(\xi, r)$ in Γ_ξ is isomorphic to the ball $B_{\Gamma^n}(a_1 a_2 \cdots a_n, r)$ in Γ^n, provided n is sufficiently large.

Theorem 5.4. *Let G be a finitely generated self-similar bounded automata group acting transitively on any C^n, then the set*

$$\{\mu \mid \Gamma_\mu \text{ is quasi-isometric to a half-line}\}$$

consists of at most two orbits.

Proof. Suppose, by contradiction, that there are at least three Schreier graphs quasi-isometric to a half-line, say corresponding to the orbits of ξ, η, ϕ.

Let $\xi = a_1 a_2 \cdots$ as before, $\eta = y_1 y_2 \cdots$ and $\phi = z_1 z_2 \cdots$, and denote by $d(-,-)$ the geodesic distance in Γ^n or in Γ_ξ. Since ξ, η and ϕ lie in different orbits, the distances between their prefixes must diverge as $n \to \infty$.

For each n, consider the vertices

$$\xi_{y,n} := y_1 y_2 \cdots y_n a_{n+1} a_{n+2} \cdots \quad \text{and} \quad \xi_{z,n} := z_1 z_2 \cdots z_n a_{n+1} a_{n+2} \cdots .$$

These vertices belong to Γ_ξ for all n, since they are cofinal with ξ (see [7]). By the definition of convergence above, and using the divergence of prefixes discussed in the previous paragraph, we have

$$d(\xi, \xi_{y,n}) \to \infty, \qquad d(\xi, \xi_{z,n}) \to \infty, \qquad d(\xi_{y,n}, \xi_{z,n}) \to \infty \quad \text{as } n \to \infty,$$

and the subgraphs induced by $\{\xi_{y,n}\}_{n\in\mathbb{N}}$ and $\{\xi_{z,n}\}_{n\in\mathbb{N}}$ inside Γ_ξ are quasi-isometric to a half-line. Consequently, Γ_ξ must have at least two ends, which yields a contradiction. □

Self-similar bounded automata groups for which almost all Schreier graphs have two ends are listed in [7]. In the case of binary alphabet X, these correspond to automata that appear in [30], including the first Grigorchuk group. To it, we can apply Proposition 5.6 which applies, more generally, to any bounded automata torsion group. To start, we need the following lemma.

Lemma 5.5. *Let G be a finitely generated bounded automata group and let η be an infinite ray in $\partial\mathcal{T}_d$. If the Schreier graph Γ_η is context-free, then the set of words one can read on geodesics in the graph starting from η is a regular language.*

Proof. We recall that, by definition, Γ_η has finitely many end-cone types. All we have to do is to construct a finite state automaton that reads geodesics. The states are of the form (C, ζ) where C is an end-cone type and ζ ranges in the frontier points of a given end-cone of type C. Note that the set of states is finite since the number of end-cone types is finite, and the number of frontier points for a given end-cone is finite too. The initial state is (C_0, η) with C_0 the end-cone type of the base vertex η and all the states are final. We then add a transition $(C_1, \zeta_1) \xrightarrow{a} (C_2, \zeta_2)$ if there exists an edge, labelled with a, from the frontier point corresponding to ζ_1 of an end-cone of type C_1, to a point that corresponds to ζ_2 which belongs to $\Gamma_\eta(\zeta_1) \setminus \Delta_\eta(\zeta_1)$. Moreover, this vertex ζ_2 is a frontier point of an end-cone of type C_2. □

Proposition 5.6. *Let G be an infinite finitely generated torsion bounded automata group, and let η be an infinite ray. If the Schreier graph of η is infinite, then* $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ *is not context-free.*

Proof. If $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is context-free, then so is the corresponding Schreier graph. Thus, from Lemma 5.5 we know that the language of all geodesics in this graph is regular. Applying the pumping lemma for regular languages (see, e.g., [29, Theorem 1.70]), we see that this language of geodesics contains some sub-language $\{xy^n z \mid n \in \mathbb{N}\}$ where $y \in X^*$ is a non-empty word. This contradicts our assumption that G is torsion, since $xy^k z$ cannot be a geodesic when k is the order of the element given by y. □

To summarise, combining Theorem 5.3 with Theorem 5.4 we show that there are finitely generated bounded automata groups such that almost surely the Schreier graphs are not context-free. An interesting example of a group where almost all Schreier graphs are one-ended trees but are not context-free, is the following.

Iterated monodromy group of $z^2 + i$ (see [7]). Using the notation introduced in Section 2, the group is generated by the three automorphisms

$$a = (b, c), \quad b = (1, 1) \cdot s, \quad c = (a, 1)$$

where $1 \neq s \in \mathrm{Sym}(\{0, 1\})$. In this case, it is easy to see that the trees are not quasi-isometric to half-lines. Indeed, if $\eta = c_1 c_2 \cdots$ is an infinite ray, the graph Γ_η contains vertices $v_k := 0^k c_{k+1} \cdots$. By an inductive argument, one can show that from each v_k there is a path to $1^k c_{k+1} \cdots$ and a different path to $1^{k-2}00c_{k+1} \cdots$. In particular, if the graph is one-ended, then it is a half-line with infinitely many paths of unbounded lengths attached, and it is easy to see that such a graph does not have finitely many end-cone types, and so is not context-free.

Our analysis above is based on the number of ends in a typical Schreier graph, that is, a Schreier graph from a subset of $\partial\mathcal{T}_d$ of measure zero. But our Theorem 4.1 concerns infinite rays that are periodic, and such rays form a subset of $\partial\mathcal{T}_d$ of measure zero. Therefore, a stronger version of the theorem where 'ET0L' would be replaced with 'context-free' might still be possible. Below, we will use Theorem 5.3 and Proposition 5.6 to provide some examples where this is not the case, as all (and not only almost all) the Schreier graphs are not context-free. A useful result here is [7, Theorem 11] which provides a criterion to determine whether all the Schreier graphs are one-ended.

Hanoi tower group on three pegs (see [16]). It is known that any Schreier graph of this group is one-ended and not quasi-isometric to the half-line (e.g. [7, Remark 3]), hence any $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is not context-free.

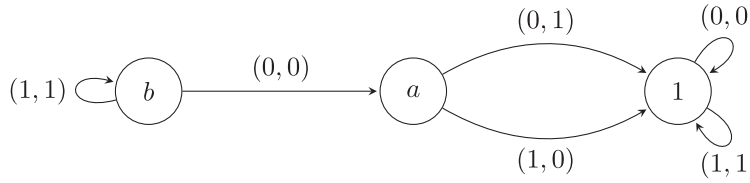

FIGURE 5 Automaton of the infinite dihedral group.

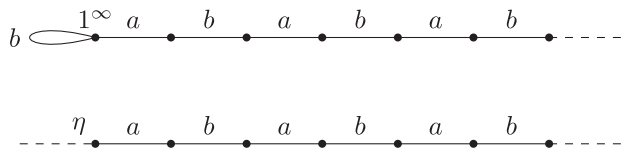

FIGURE 6 Isomorphism classes of Schreier graphs in the infinite dihedral group.

Basilica group. We can directly apply the theorem to show that the language $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ cannot be context-free for the Basilica group, since its Schreier graphs are fully classified in [10]. Namely, all Schreier graphs are one-ended, two-ended or four-ended. In fact, the latter case is a single exception. If the Schreier graph has one or two ends, by the classification it is not quasi-isometric to a tree. On the other hand, for the case of the four-ended graph, it is clear that it does not have finitely many end-cone types (see [10, Theorem 4.6 and fig. 7]).

First Grigorchuk group. In this example almost all graphs are two-ended graphs. Namely, it has just one one-ended Schreier graph, the one containing 1^∞, the rightmost point in the boundary. In this case, we use Proposition 5.6 to conclude the non-context-freeness.

We end the section with two examples. One, for which Theorem 4.1 can indeed be strengthened, and $\mathsf{WP}(G, X, Stab(\eta))$ is context-free. And one where we do not know whether $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is context-free.

Infinite dihedral group. This group can be seen as the self-similar group generated by the automorphisms

$$a = (1, 1) \cdot s, \quad b = (a, b),$$

where $1 \neq s \in \mathrm{Sym}(\{0, 1\})$ or, equivalently by the bounded automaton in Figure 5. One Schreier graph is one-ended (containing 1^∞), while all the others are isomorphic to the Cayley graph of $\mathbb{Z}$, an infinite line, see Figure 6. Thus, all the Schreier graphs are context-free.

Grigorchuk group $G_{\overline{01}}$ (see, e.g., [15]). It is well-known that this is a self-similar bounded automata group containing non-torsion elements and all the Schreier graphs are quasi-isometric to a line or a half-line. However, we do not know whether such graphs are context-free or not.

6 | FURTHER RESEARCH

In this paper, we showed that in a finitely generated group $G \leqslant \mathrm{Aut}(\mathcal{T}_d)$ the membership problem $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ in stabilisers of infinite eventually periodic rays $\eta \in \partial\mathcal{T}_d$ is an ET0L language. It turns out that the word problem $\mathsf{WP}(G, X)$ can be characterised in terms of these languages, as explained in the following proposition.

Proposition 6.1. *The word problem of a group $G \leqslant \mathrm{Aut}(\mathcal{T}_d)$ coincides with the intersection of all* $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ *with η periodic.*

Proof. It is straightforward that any $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ contains the word problem. On the other side, the action on $\mathcal{T}_d$ is faithful. So, if an element stabilises all the vertices of the tree, then it is the identity. Now observe that if $w \in \mathsf{WP}(G, X, \mathrm{Stab}(\eta))$, then w stabilises all the prefixes of η. Take w in the intersection of all $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ with η periodic. This means that w stabilises all possible finite words in C^* and hence it is the identity. □

The immediate corollary that the word problem of a bounded automata group is an intersection of infinitely many ET0L languages does not in itself say much, as it is well-known that any language is an intersection of (infinitely many) regular languages. But it motivates the following natural question.

Question 6.2. Is it true that the word problem of a bounded automata group is a finite intersection of ET0L languages?

In Section 5, we proved that the languages $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ are not context-free under some additional hypothesis. We also mentioned there one example, a non-torsion group from Grigorchuk's family, $G_{\overline{01}}$, for which we do not know whether these languages are context-free or not. We think, it is not, and we ask the following.

Question 6.3. Is there a non-virtually free self-similar bounded automata group with $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ context-free?

We are also interested to know if η is computable from $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$. That is, if the language uniquely determines the ray.

Question 6.4. Let G be the first Grigorchuk group, and let η be the word $0101^2 01^3 01^4 \ldots$. Is it true that the subgroup membership problem $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ is not ET0L?

The statements Theorems 3.12 and 4.1 put together give us a characterisation of the generating function for $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$, when η is an eventually periodic infinite word in the alphabet C. There are certain subclasses of *indexed languages* that have known characterisations of their generating functions with potential closed-form expressions (see [1]).

Question 6.5. For what bounded automata group and infinite rays does the language $\mathsf{WP}(G, X, \mathrm{Stab}(\eta))$ belong to the subclasses of indexed languages as studied in [1]?

ACKNOWLEDGEMENTS

The first-named author and the fifth-named author acknowledge support from the Swiss Government Excellence Scholarship. The first-, fourth- and fifth-named authors acknowledge support from Swiss NSF Grant 200020-200400. The second-, third-, fifth- and sixth-named authors are members of the Gruppo Nazionale per le Strutture Algebriche, Geometriche e le loro Applicazioni (GNSAGA) of the Istituto Nazionale di Alta Matematica (INdAM). The third-named author is also a member of the PRIN 2022 'Group theory and its applications' research group and gratefully acknowledges the support of the PRIN project 2022-NAZ-0286, funded by the European

Union - Next Generation EU, Missione 4 Componente 1 CUP B53D23009410006, PRIN 2022 - 2022PSTWLB - Group Theory and Applications. The third-named author also gratefully acknowledges the support of the Università degli Studi di Milano–Bicocca (FA project 2021-ATE-0033 'Strutture Algebriche'). The fifth-named author acknowledges support from the Grant QUALIFICA by Junta de Andalucía grant number QUAL21 005 USE and from the research grant PID2022-138719NA-I00 (Proyectos de Generación de Conocimiento 2022) financed by the Spanish Ministry of Science and Innovation. The first-named author thanks the Dipartimento di Matematica e Applicazioni of the Università di Milano-Bicocca for their hospitality. The fifth-named author also thanks the Section de mathématiques of the Université de Genève for their hospitality. The authors are grateful to Murray Elder for helpful discussions and to the anonymous referees for useful comments on the first draft of the paper.

Open access publishing facilitated by Universite de Geneve, as part of the Wiley - Universite de Geneve agreement via the Consortium Of Swiss Academic Libraries.

JOURNAL INFORMATION

The *Journal of the London Mathematical Society* is wholly owned and managed by the London Mathematical Society, a not-for-profit Charity registered with the UK Charity Commission. All surplus income from its publishing programme is used to support mathematicians and mathematics research in the form of research grants, conference grants, prizes, initiatives for early career researchers and the promotion of mathematics.

ORCID

Davide Perego https://orcid.org/0009-0008-5939-7942

REFERENCES

1. J. Adams, E. Freden, and M. Mishna, *From indexed grammars to generating functions*, RAIRO Theor. Inform. Appl. **47** (2013), no. 4, 325–350.
2. A. V. Anisimov, *The group languages*, Kibernetika (Kiev) **7** (1971), no. 4, 18–24.
3. P. R. J. Asveld, *Controlled iteration grammars and full hyper-AFL's*, Inf. Control **34** (1977), no. 3, 248–269.
4. J. Berstel and D. Perrin, *Theory of codes*, vol. 117, Academic Press, Cambridge, MA, 1985.
5. A. Bishop and M. Elder, *Bounded automata groups are co-ET0L*, Language and automata theory and applications, Lecture Notes in Comput. Sci., vol. 11417, Springer, Cham, 2019, pp. 82–94.
6. I. Bondarenko, *Groups generated by bounded automata and their Schreier graphs*, Ph.D. thesis, Texas A&M University, 2007.
7. I. Bondarenko, D. D'Angeli, and T. Nagnibeda, *Ends of Schreier graphs and cut-points of limit spaces of self-similar groups*, J. Fractal Geom. **4** (2017), no. 4, 369–424.
8. L. Ciobanu and M. Elder, *Solutions sets to systems of equations in hyperbolic groups are EDT0L in PSPACE (Track B: Automata, Logic, Semantics, and Theory of Programming)*. In 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019). Leibniz International Proceedings in Informatics (LIPIcs), vol. 132, pp. 110:1–110:15, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.
9. L. Ciobanu, M. Elder, and M. Ferov, *Applications of L systems to group theory*, Internat. J. Algebra Comput. **28** (2018), no. 2, 309–329.
10. D. D'Angeli, A. Donno, M. Matter, and T. Nagnibeda, *Schreier graphs of the Basilica group*, J. Mod. Dyn. **4** (2010), no. 1, 167–205.
11. M. Dehn, *Über unendliche diskontinuierliche Gruppen*, Math. Ann. **71** (1911), no. 1, 116–144.
12. V. Diekert and M. Elder, *Solutions of twisted word equations, EDT0L languages, and context-free groups*. In 44th International Colloquium on Automata, Languages, and Programming (ICALP 2017). Leibniz International Proceedings in Informatics (LIPIcs), vol. 80, pp. 96:1–96:14, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

13. V. Diekert and A. Muscholl, *Solvability of equations in free partially commutative groups is decidable*, Automata, languages and programming, Lecture Notes in Comput. Sci., vol. 2076, Springer, Berlin, 2001, pp. 543–554.
14. A. Evetts and A. Levine, *Equations in virtually abelian groups: languages and growth*, Internat. J. Algebra Comput. **32** (2022), no. 03, 411–442.
15. R. Grigorchuk, *Degrees of growth of finitely generated groups and the theory of invariant means*, Izv. Akad. Nauk SSSR Ser. Mat. **48** (1984), no. 5, 939–985.
16. R. Grigorchuk and Z. Šunić, *Asymptotic aspects of Schreier graphs and Hanoi towers groups*, C. R. Math. Acad. Sci. Paris **342** (2006), no. 8, 545–5500.
17. M. Lohrey, *Membership problems in infinite groups*. In Twenty years of theoretical and practical synergies, Springer Nature Switzerland, 2024, pp. 44–59.
18. B. Mohar and W. Woess, *A survey on spectra of infinite graphs*, Bull. Lond. Math. Soc. **21** (1989), no. 3, 209–234.
19. D. E. Muller and P. E. Schupp, *Groups, the theory of ends, and context-free languages*, J. Comput. System Sci. **26** (1983), no. 3, 295–310.
20. V. Nekrashevych, *Self-similar groups*, Mathematical Surveys and Monographs, vol. 117, Amer. Math. Soc., Providence, RI, 2005.
21. M. Nielsen, *E0L systems with control devices*, Acta Informat. **4** (1975), no. 4, 373–386.
22. R. Pemantle and M. Wilson, *Analytic combinatorics in several variables*, Cambridge University Press, Cambridge, 2015.
23. E. Rodaro, *Generalizations of the Muller–Schupp theorem and tree-like inverse graphs*, J. Lond. Math. Soc. **109** (2024), no. 5, e12903.
24. G. Rozenberg, *Extension of tabled* OL*-systems and languages*, Internat. J. Comput. Information Sci. **2** (1973), 311–336.
25. G. Rozenberg and A. Salomaa, *The book of L*, Springer, Berlin Heidelberg, 1986.
26. J. Sakarovitch, *Elements of automata theory*, Cambridge University Press, Cambridge, 2009.
27. A. Salomaa, *CHAPTER 3 - Formal Languages and Power Series*, Editor(s): J. Van Leeuwen, In Handbook of Theoretical Computer Science, Formal Models and Semantics, Elsevier, 1990, pp. 103–132.
28. S. Sidki, *Automorphisms of one-rooted trees: growth, circuit structure, and acyclicity*, J. Math. Sci. (New York) **100** (2000), no. 1, 1925–1943.
29. M. Sipser, *Introduction to the theory of computation*, 3rd ed., Course Technology, Boston, MA, 2013.
30. Z. Šunić, *Hausdorff dimension in a family of self-similar groups*, Geom. Dedicata **124** (2007), 213–236.
31. J. van Leeuwen, *Variations of a new machine model*, 17th Annual Symposium on Foundations of Computer Science (Houston, TX, 1976), IEEE Comput. Soc., Long Beach, CA, 1976, pp. 228–235.