Doctor of Philosophy
PhD Thesis: Mathematics

Dissertation

Geodesic Languages and Co-Word Problems over Groups

Alex Bishop

December 2021

Supervisor: Professor Murray Elder

School of Mathematical and Physical Sciences
Faculty of Science
University of Technology Sydney

Certificate of Original Authorship

I, Alex Bishop, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Science at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note:
Signature removed prior to publication.

Date: 22 December 2021

Abstract

In 1968, Milnor asked if a finitely-generated group could have volume growth that is neither exponential nor polynomial (so-called 'intermediate'), and if there is an algebraic classification of groups with polynomial volume growth. We consider the analogous questions for geodesic growth.

We show that no virtually abelian group can have intermediate geodesic growth. In particular, we completely characterise the geodesic growth for every virtually abelian group. We show that the geodesic growth is either polynomial of an integer degree with rational geodesic growth series, or exponential with holonomic geodesic growth series. In addition, we show that the language of geodesics is blind multicounter. These results hold for each finite weighted monoid generating set of any virtually abelian group.

A direct consequence of Gromov's classification of polynomial volume growth is that if a group has polynomial geodesic growth with respect to some finite generating set, then it is virtually nilpotent. Until now, the only known examples with polynomial geodesic growth were all virtually abelian. We furnish the first example of a virtually 2-step nilpotent group having polynomial geodesic growth with respect to a certain finite generating set.

Holt and Röver proved that finitely-generated bounded automata groups have indexed co-word problems. We sharpen this result to show that their co-word problem is ET0L. We do so using an equivalent machine model known as a cspd automaton. This extends a result of Ciobanu, Elder and Ferov who showed this for the first Grigorchuk group by explicitly constructing an ET0L grammar.

Acknowledgements

Firstly, I would like to thank my supervisor Murray Elder for introducing me to these research problems, for his support throughout my Honours and PhD, and for his helpful comments when revising my thesis. I also thank Michal Ferov for his feedback and suggestions when revising my thesis.

I thank Moyna Ward for giving me a room to stay at the beginning of my PhD. I would like to sincerely thank my parents, my brother, and Jessica Chen for their support throughout my studies.

Contents

Chapter 1

Introduction

Two of the most important results in geometric group theory are Gromov's classification of groups with *polynomial volume growth* [56], and Grigorchuk's construction of a group with *intermediate volume growth* [53]. These results answer two questions originally posed by Milnor, namely, whether the *volume growth* of groups must always be either exponential, or polynomial of an integer degree; and if there is an algebraic classification of groups with polynomial volume growth [77]. In this thesis, we consider the analogous questions for *geodesic growth*.

The *volume growth function* of a group counts the number of elements that can be represented using words up to a given length. It was shown by Gromov that a group has polynomial volume growth if and only if it is *virtually nilpotent* [56]. The *geodesic growth function* of a group counts the number of *geodesic words* (i.e. minimal-length representatives of group elements) with a given upper bound on their lengths. The *volume (resp. geodesic) growth series* is then the power series whose coefficients are the values of the volume (resp. geodesic) growth function. Since each element of a group has at least one corresponding geodesic, we see that the geodesic growth is bounded from below by the volume growth. From this and Gromov's theorem, we see that only *virtually nilpotent* groups may have polynomial geodesic growth. It is well known that the class of *nilpotent* groups come in a sequence of *steps*, the first *step* being the abelian groups. Thus, to obtain a classification of polynomial geodesic growth, it is natural to start with the *virtually abelian* groups.

The study of geodesic growth for abelian groups began in 1997 when Shapiro considered the function $p_S \colon G \to \mathbb{N}$ which counts the geodesics corresponding to a given element of a group G [90]. The function p_S is referred to as the *Pascal function* as, in the case of free-abelian groups, it resembles a Pascal triangle. The geodesic growth of virtually abelian groups was considered by Bridson, Burillo, Elder and Šunić who provided the first example of a group with polynomial geodesic growth that is not *virtually cyclic* [19]. Moreover, they provided a sufficient condition for a virtually abelian group to have polynomial geodesic growth with respect to some generating set, a condition for a group to have exponential geodesic growth with respect to every generating set, and proved

that the geodesic growth function of a virtually cyclic group is either exponential, or polynomial of an integer degree. In this thesis, we extend their results by completely characterising the geodesic growth for virtually abelian groups.

Theorem A. *Let G be a virtually abelian group with a finite weighted monoid generating set S. Then the geodesic growth with respect to S is either polynomial of integer degree with rational geodesic growth series, or exponential with holonomic geodesic growth series.*

We provide the first example of a group with polynomial geodesic growth that is not virtually abelian. In particular, we show that the *virtually Heisenberg group*

$$H_3 \rtimes C_2 = \langle a, b, c, t \mid [a, b] = c,\ [a, c] = [b, c] = t^2 = 1,\ a^t = b \rangle$$

has a polynomial upper bound on its geodesic growth function with respect to the generating set $S = \{a, a^{-1}, t\}$. We prove this result in Theorem B. This example shows that a classification of polynomial geodesic growth is more complicated than just a subclass of virtually abelian groups.

Theorem B. *The geodesic growth function of $H_3 \rtimes C_2$ with respect to $S = \{a, a^{-1}, t\}$ is bounded from above by a polynomial of degree* 8.

It was shown by Duchin and Shapiro that the volume growth series of the Heisenberg group H_3 is rational with respect to every generating set [39]. Theorem A shows us that if the geodesic growth of a virtually abelian group is polynomial, then its geodesic growth series is rational. However, it is not clear if this holds for the geodesic growth function of our virtually Heisenberg example $H_3 \rtimes C_2$. In fact, computational experiments suggest that the geodesic growth series is not rational (see [12]). Thus, this may be the first example with polynomial geodesic growth and non-rational geodesic growth series.

The question of the existence of a group with intermediate geodesic growth was considered as early as 1993 by Grigorchuk and Shapiro (see [54, p. 756]). In the PhD thesis of Brönnimann most of the groups known to have intermediate volume growth, at the time, were shown to have exponential geodesic growth with respect to their standard generating sets [21, Chapter 3]; these results were an extension of an unpublished work of Elder, Gutierrez and Šunić where this was shown only for the *first Grigorchuk group* [41]. We cannot, at this time, eliminate the possibility of there being a virtually nilpotent group with intermediate geodesic growth. Thus, the results in this thesis also have application to the search of a group with intermediate geodesic growth.

Most of the literature on geodesic growth has been concerned with either showing that the language of geodesics is regular (see Section 2.1), or that the geodesic growth series is rational (see Section 2.2.1) with respect to some particular generating sets. It is known that the language of geodesics for a *hyperbolic group* is regular with respect to any finite generating set [43, Theorem 3.4.5]. This result was generalised by Neumann and Shapiro to any group and generating set with the *falsification by fellow traveller property* [80, Proposition 4.2 on p. 267]. There are many results for particular generating sets of certain *Coxeter groups*, *Artin groups* [3, 4, 6, 33, 60, 67, 75], and *Garside groups* [24, 88]. Shapiro studied the Pascal function for abelian and hyperbolic groups [90]. It was shown by

Loeffler, Meier and Worthington that having a regular language of geodesics is preserved by graph product [74]. Hermiller, Holt and Rees studied groups whose languages of geodesics are *locally testable* (which is a proper subfamily of the regular languages) [59]. Cleary, Elder and Taback showed that the language of geodesics for the *lamplighter group* is *context-free* and *counter* for some generating sets; and that the language of geodesics for *Thompson's group F* is not regular for any generating set [34].

We prove Theorem A by constructing an algorithm that converts geodesics into *patterned words*. From this algorithm, we find a bijection from the set of geodesics of a virtually abelian group to a certain formal language with a *holonomic* growth series. It is then natural to ask if there is a formal-language characterisation for the language of geodesics for a virtually abelian group. In Theorem C, we obtain such a characterisation by implementing this algorithm using *blind multicounter automata*.

Theorem C. *The language of geodesics of a virtually abelian group with respect to any finite weighted monoid generating set S is blind multicounter.*

A *formal language over a group* is a set of words whose letters are taken from the generating set. So far we have discussed our results on the language of geodesics for a group. In this thesis, we also study the *co-word problem*, that is, the language of words that do not correspond to the group identity. Characterisations of such languages provide us with one measure for the computational difficulty involved with computing in a group.

The *word problem* of a group G with respect to a finite monoid generating set S, denoted WP_S, is the set of all words in S^* that correspond to the group identity. We see that the word problem completely specifies a group as $\langle S \mid \mathrm{WP}_S \rangle$ is a presentation for G. This formal language is one characterisation of the difficulty of computing within a group, i.e., checking if two words $u, v \in S^*$ represent the same group element is equivalent to checking if the word uv^{-1} is in the word problem WP_S. The *co-word problem* of a group, denoted coWP_S, is the complement of the word problem in the sense that $\mathrm{coWP}_S = S^* \setminus \mathrm{WP}_S$.

There are groups for which it is not possible to decide membership to the word problem. Interestingly, there are such groups for which the word problem is *recursively enumerable* but not *recursive*, that is, there is an algorithm that lists every word in the word problem (with no guarantee of order) but no such algorithm which lists out every word in the co-word problem. In particular, there are finitely-presented groups with unsolvable word problems. The existence of such examples was shown independently by Novikov [81] (see [20] for an English translation) and Boone [18]. An explicit example with 10 generators and 27 relators was given by Collins [35], then later a simpler example with 2 generators and 27 relators was given by Wang, Li, Yang and Lin [94]. We see that the word problem for any finitely-presented group is recursively enumerable (see Proposition A.1).

Suppose that $\mathcal{F}$ is a family of formal languages that is closed under *inverse word homomorphism*. Then, if the word or co-word problem with respect to some generating set belongs to $\mathcal{F}$, it belongs to $\mathcal{F}$ for all generating sets (see Lemma 3.2). Thus, for such a family it is well defined to state that a group has a word or co-word problem in $\mathcal{F}$. Examples of such families are the *regular*, *context-free* and *context-sensitive* languages; the family of *ET0L languages* which are a type of L-system, introduced by Rozenberg [87], that generalises context-free languages, and form a subfamily of context-sensitive languages;

and the family of *blind multicounter languages* [52] which generalise counter languages and are equivalent to the class of *reversal bounded multicounter languages* [8] and *Parikh languages* [64]. If $\mathcal{F}$ is a family of formal languages that is closed under inverse word homomorphism, and the word (resp. co-word) problem of a group lies in $\mathcal{F}$, then we say that the group is an $\mathcal{F}$ (resp. co-$\mathcal{F}$) group.

It is interesting to find classifications of groups whose languages WP_S and coWP_S belong to certain language families. The study of this problem began with Anisimov who showed that a group is finite if and only if WP_S is a regular language [2]. Further classifications were obtained by Muller and Schupp who showed that a group is *virtually free* if and only if its word problem is a context-free language [78]; and Elder, Kambites and Ostheimer who showed that a group is virtually abelian if and only if its word problem is a blind multicounter language [42].

The study of the group co-word problem, coWP_S, gives us an additional source of such classifications. The class of groups for which coWP_S is context-free was first studied by Holt, Rees, Röver and Thomas [61]. Their results were then extended by Lehnert and Schweitzer [70] who showed that Thompson's group V has a context-free co-word problem. Combining a result of Bleak, Matucci and Neunhöffer [17] with a remark in Lehnert's thesis [71, § 4.2], it is conjectured that every group with context-free co-word problem is a subgroup of Thompson's group V. Moreover, it is conjectured that Grigorchuk's group does not have a context-free co-word problem [17].

The class of *bounded automata groups* includes important examples such as Grigorchuk's group of intermediate growth, the Gupta-Sidki groups, and many more [55, 57, 79, 91]. It was shown by Holt and Röver [63] that bounded automata groups have *indexed* co-word problems. ET0L languages form a proper subfamily of the indexed languages introduced by Aho [1] (see Corollary 4.1 in [36] and Proposition 4.5 in [40]). For the case of Grigorchuk's group, it was later shown by Ciobanu, Elder and Ferov that the co-word problem is ET0L [32]. They proved this result by explicitly constructing an ET0L grammar to recognise the co-word problem. In Theorem D we use an equivalent machine model to generalise this result to all bounded automata groups.

Theorem D. *Every finitely-generated bounded automata group is co-ET0L.*

All results in this thesis are with respect to monoid generating sets for groups. In particular, this means that we do not assume that our generating sets are *symmetric*. For example, the group of integers $\mathbb{Z} = \langle a \mid - \rangle$ is generated by $\{a^{-1}, a^3\}$. Moreover, we prove Theorems A and C for each finite weighted monoid generating set. That is, for each generator we associate a positive integer weight, and we say that a word is a geodesic if it represents an element with minimal weight. Notice that we may recover the usual definition of a geodesic by choosing the weight of each generator to be one.

1.1. Structure

This thesis is structured as follows. In Chapter 2, we provide background on formal language theory and define the families of formal languages that are used in our proofs.

In Chapter 3 we study the co-word problem for bounded automata groups, and prove Theorem D. We then return to formal language theory in Chapter 4 where we define the family of polyhedrally constrained languages. Then, in Chapter 5 we prove provide a characterisation of the language of geodesics and geodesic growth of virtually abelian groups by proving Theorems A and C. Finally, in Chapter 6 we consider virtually nilpotent groups and prove Theorem B.

1.2. Attribution of Results

Theorems A and C are published in the single-authored paper [13]. Theorem B is joint work with my supervisor, Murray Elder, a preprint of this work is available in [14]. Theorem D is also joint work with Elder and has appeared in the conference proceedings of LATA (Language and Automata Theory and Applications) 2019 [15].

1.3. Notation

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ denote the set of nonnegative integers, including zero, and $\mathbb{N}_+ = \{1, 2, 3, \ldots\}$ the set of positive integers.

Let G be a group, and let $g, h \in G$, then $[g, h] = ghg^{-1}h^{-1}$ and $g^h = hgh^{-1}$. Given two subgroups $H, K \leqslant G$, we write $[H, K]$ for the subgroup

$$[H, K] = \langle \{[h, k] \mid h \in H,\ k \in K\} \rangle .$$

A group G is *k-step nilpotent* if there is a finite sequence of subgroups

$$G_0 = G,\ G_1 = [G, G_0],\ G_2 = [G, G_1],\ G_3 = [G, G_2],\ \ldots,\ G_k = [G, G_{k-1}] = \{1\}.$$

Notice that the 1-step nilotent groups are precisely the abelian groups. Moreover, for each k, the set of $(k + 1) \times (k + 1)$ invertible upper-triangular integer matrices with 1's on their diagonal form a k-step nilpotent group. In fact, it is a result of Auslander that each nilpotent (or more generally each *polycyclic*) group has a faithful representation in $\mathrm{SL}(n, \mathbb{Z})$ for some n [7, Theorem 2].

Let $\mathcal{P}$ be some group property, e.g., being abelian, nilpotent, or free. Then, we say that a group is *virtually* $\mathcal{P}$ is it has a finite-index subgroup with the property $\mathcal{P}$. For example, we say that a group is virtually nilpotent if it has a finite-index nilpotent subgroup.

Let G be a group with a finite generating set S, then we write S^* for the set of all words, including the empty word $\varepsilon \in S^*$, in the letters of S; and $\overline{\sigma} \in G$ for the group element corresponding to the word $\sigma \in S^*$. We endow S with a weighting, that is, for each generator $s \in S$ we assign a positive integer weight $\omega(s) \in \mathbb{N}_+$. We then say that S is a finite weighted generating set for the group G. The *weight* of a word $\sigma = \sigma_1\sigma_2 \cdots \sigma_k \in S^*$ is then given by $\omega(\sigma) = \sum_{i=1}^{k} \omega(\sigma_i)$. Moreover, we write $|\sigma|_S = k$ for the *word length* of σ. The *weighted length* of an element $g \in G$ is then defined as the minimum weight required to represent it as a word, that is,

$$\ell_S(g) = \min\{\omega(\sigma) \mid \overline{\sigma} = g \text{ where } \sigma \in S^*\}.$$

We may now define the volume growth function $a_S\colon \mathbb{N} \to \mathbb{N}$ as follows.

Definition 1.1. *The* volume growth function $a_S\colon \mathbb{N} \to \mathbb{N}$ *is defined as*

$$a_S(n) = \#\{g \in G \mid \ell_S(g) \leqslant n\}.$$

That is, $a_S(n)$ counts the elements which can be represented by a word of length n or less.

We say that a word $\sigma \in S^*$ is a *geodesic* if it represents $\overline{\sigma}$ with minimal weight, that is, if $\omega(\sigma) = \ell_S(\overline{\sigma})$. We write Geod_S for the set of all geodesic words with respect to the generating set S, that is,

$$\mathrm{Geod}_S = \{\sigma \in S^* \mid \omega(\sigma) = \ell_S(\overline{\sigma})\}.$$

We then define the *geodesic growth function* $\gamma_S\colon \mathbb{N} \to \mathbb{N}$ as follows.

Definition 1.2. *The* geodesic growth function $\gamma_S\colon \mathbb{N} \to \mathbb{N}$ *is defined as*

$$\gamma_S(n) = \#\{\sigma \in \mathrm{Geod}_S \mid \omega_S(\sigma) \leqslant n\}$$

This function counts the number of geodesic words of length n or less.

Notice that the volume and geodesic growth functions can be at most exponential as

$$a_S(n) \leqslant \gamma_S(n) \leqslant \sum_{i=0}^{n} |S|^i \leqslant |S|^{n+1}.$$

We say that a (volume/geodesic) growth function $f\colon \mathbb{N} \to \mathbb{N}$ has

- *polynomial growth* if there is some $\beta, d \in \mathbb{N}_+$ such that $f(n) \leqslant \beta n^d$ for each $n \geqslant 1$;
- *exponential growth* if there is an $\alpha \in \mathbb{R}$ with $\alpha > 1$ such that $f(n) \geqslant \alpha^n$; and
- *intermediate growth* if its growth is neither polynomial nor exponential.

Notice that the volume and geodesic growth functions are submultiplicative, that is, if $f\colon \mathbb{N} \to \mathbb{N}$ is a growth function, then $f(n+m) \leqslant f(n)f(m)$ for each $n, m \in \mathbb{N}$. Thus, we may apply the following result.

Lemma 1.3 (Fekete's lemma [47])**.** *If $f\colon \mathbb{N} \to \mathbb{N}$ is submultiplicative, then the* growth rate $\alpha_f = \lim_{n\to\infty} \sqrt[n]{f(n)}$ *is defined.*

From Lemma 1.3, we see that a function $f\colon \mathbb{N} \to \mathbb{N}$ has exponential growth if and only if the growth rate $\alpha_f > 1$.

In this thesis, we are interested in studying the asymptotics of growth functions by considering their associated generating functions. We write A_S and Γ_S for the generating functions associated with a_S and γ_S, respectively.

Definition 1.4. *We write*

$$A_S(z) = \sum_{n=0}^{\infty} a_S(n) z^n \quad \textit{and} \quad \Gamma_S(z) = \sum_{n=0}^{\infty} \gamma_S(n) z^n$$

for the volume *and* geodesic growth series*, respectively.*

We write $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ for a finite list of variables. Then, for each vector $\mathbf{n} = (n_1, n_2, \ldots, n_m)$, we then write $\mathbf{x}^{\mathbf{n}} = x_1^{n_1} x_2^{n_2} \cdots x_m^{n_m}$. We may write a multivariate generating series as $f(\mathbf{x}) = \sum_{\mathbf{n} \in \mathbb{N}^m} c_{\mathbf{n}} \mathbf{x}^{\mathbf{n}}$ where each $c_{\mathbf{n}}$ is a constant. We write $\mathbb{C}[[\mathbf{x}]]$, $\mathbb{C}[\mathbf{x}]$, $\mathbb{C}((\mathbf{x}))$, and $\mathbb{C}(\mathbf{x})$ for the class of formal power series, polynomials, formal Laurent series, and rational functions, respectively, over the variables $\mathbf{x} = (x_1, x_2, \ldots, x_m)$. Moreover, we write $\partial_{x_i} f(\mathbf{x})$ for the formal partial derivative of $f(\mathbf{x})$ with respect to x_i. We use this notation in Section 2.2 where we define multivariate generating functions and the classes of rational, algebraic and holonomic power series.

Let G be a group with finite monoid generating set S, then the *word* and *co-word* problem of G with respect to S are given as

$$\mathrm{WP}(G, S) = \{w \in S^* \mid \overline{w} = 1_G\} \quad \text{and} \quad \mathrm{coWP}(G, S) = \{w \in S^* \mid \overline{w} \neq 1_G\},$$

respectively. Notice here that $\mathrm{coWP}(G, S) = S^* \setminus \mathrm{WP}(G, S)$, that is, the two sets are complements of each another with respect to the set S^*.

Chapter 2

Formal Languages and Generating Functions

How can we describe and study the complexity of combinatorial structures? One answer is to use the theory of *formal languages*, that is, after finding a bijection from our combinatorial structures to *words* in a formal language, we may produce generating functions and computational descriptions.

A formal language is a set of words whose letters are taken from a finite set of abstract symbols Σ known as an *alphabet*, or equivalently, a formal language is a subset of the free monoid Σ^*. We collect formal languages into families, and study the computational complexity and the generating functions of languages in these families.

In this chapter, we begin by recalling some basic definitions in formal language theory, in particular, we recall the Chomsky hierarchy in Section 2.1, and formal language generating functions in Section 2.2. In Section 2.2 we describe the classes of *rational*, *algebraic*, and *holonomic* generating functions. For each such class of generating functions we provide a family of formal languages with generating functions lying in the class, and an explicit example of such a language. We then define several particular language families which we require for the proofs and results in this thesis. In particular, we define the family of *constrained languages*, *blind multicounter languages*, and *ET0L languages*.

In Section 2.3, we define *constrained languages* with a focus on the family of *linearly constrained languages* introduced by Massazza [76]. It was shown by Massazza [76, Theorem 2] that linearly constrained languages have holonomic generating functions. We return to constrained languages in Chapter 4 where we define the family of *polyhedrally constrained languages*. This family of languages is used in the proof of Theorem A.

In Section 2.4 we study *blind multicounter automata*. We say that a language is *blind multicounter* if it is recognised by a blind multicounter automaton. In Theorem C, we show that the language of geodesics for a virtually abelian group, with respect to any finite weighted monoid generating set, is blind multicounter.

Lastly, in Section 2.5 we study the family of *ET0L languages*. ET0L languages and their deterministic counterpart, EDT0L, arise naturally in many areas of group theory [29–32,

37, 45]. In Chapter 3 we see that this family is relevant to the well-studied class of groups known as *bounded automata*. In particular, we show that the co-word problem for bounded automata groups is ET0L. Our proof relies on a machine model, known as a *cspd* automaton, which is equivalent to the family of ET0L languages. In Section 2.5 we provide a self-contained proof of an equivalence between ET0L languages and cspd automata. We prove Theorem D by constructing a cspd automaton for the co-word problem of a bounded automata group.

2.1. The Chomsky Hierarchy

The Chomsky hierarchy consists of four well-known families of languages which can be described by formal grammars with increasingly restrictive rules. In particular, the hierarchy comprises the families of *recursively enumerable*, *context-sensitive*, *context-free* and *regular languages*. Each family in the hierarchy has an equivalent machine model, which are arbitrary *Turing machines*, *linearly bounded automata*, *pushdown automata*, and *finite-state automata*, respectively. Moreover, the families in this hierarchy form a sequence of strict containment as seen in Figure 2.1.

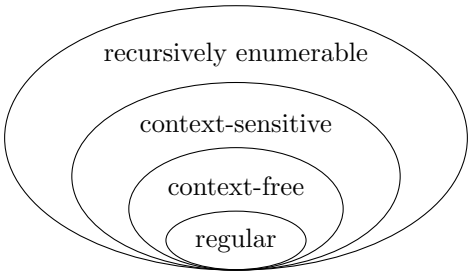

Figure 2.1: The Chomsky hierarchy.

The standard reference for this hierarchy is the 1959 paper by Chomsky [27], in which these languages were studied with respect to the complexity of their corresponding *formal grammars*. The class of Turing machines and family of recursively-enumerable languages were defined by Turing [93] in 1936, and Post [82] in 1943, respectively. These concepts were shown to be equivalent in 1947 by Post [83]. The family of regular languages and class of finite-state automata were defined and shown to be equivalent in 1956 by Kleene [66] where they were studied in the context of *nerve nets*. The family of context-free languages was shown to be equivalent to the class of pushdown automata independently by Chomsky [25] in 1962 and Evey [46] in 1963. Lastly, in 1964 it was shown by Kuroda [68] that the family of context-sensitive languages is equivalent to the class of languages recognised by a Turing machine with linearly bounded tape, that is, a Turing machine whose *work tape* (i.e. its memory) can only be linear in size with respect to the size of its input. For a more detailed history the reader is directed to [51].

In Section 2.1.1 we define each family in the Chomsky hierarchy in terms of their grammars, and describe what it means for a context-free language to be unambiguous.

Then, in Section 2.1.2 we define the class of finite-state automata. The concepts described in this section are preliminaries to our discussion of other families of languages discussed in this chapter, and are assumed in many of the proofs within this thesis.

2.1.1. Formal Grammars

A *formal grammar* is a finite set of *replacement rules* which describe how to build words in a language. There are many equivalent definitions of *formal grammar* in the literature. We define a *formal grammar* to be a 4-tuple of the form (Σ, V, S, P) where

- Σ is the alphabet of the language generated by the grammar;
- V is a finite set of *non-terminal letters* which are disjoint from the letters in Σ;
- $S \in V$ is the *starting symbol*; and
- P is a finite set of replacement rules of the form $r\colon p \to q$ where $p, q \in (V \cup \Sigma)^*$.

For each replacement rule $r\colon p \to q$ and each word $\alpha p \beta$ where $\alpha, \beta \in (V \cup \Sigma)^*$, we may write $\alpha p \beta \to^r \alpha q \beta$, that is, our rule r allows us to replace any instance of p with q. For each sequence of replacement rules $\rho = r_1 r_2 \cdots r_k \in P^*$, we write $w \to^\rho \sigma$ if there is a finite sequence of words $w_1, w_2, \ldots, w_{k-1} \in (V \cup \Sigma)^*$ such that

$$w \to^{r_1} w_1 \to^{r_2} w_2 \to^{r_3} \cdots \to^{r_k} \sigma.$$

The language generated by a formal grammar (Σ, V, S, P) is then

$$L(\Sigma, V, S, P) = \{w \in \Sigma^* \mid S \to^\rho w \text{ where } \rho \in P^*\}.$$

We classify formal grammars into four types based on the complexity of their replacement rules, in particular, every formal grammar is *Type 0*, and a grammar is

- *Type 1* if each replacement rule has the form $\alpha A \beta \to \alpha \gamma \beta$ where $A \in V$ is a non-terminal, and $\alpha, \beta, \gamma \in (V \cup \Sigma)^*$ are words;
- *Type 2* if each rule has the form $A \to \alpha$ where $A \in V$ and $\alpha \in (V \cup \Sigma)^*$; and
- *Type 3* if each rule has the form $A \to \alpha$ or $A \to \alpha B$ where $A, B \in V$ and $\alpha \in \Sigma^*$.

The classes of Type 0, 1, 2 and 3 formal grammars correspond to the families of recursively enumerable, context-sensitive, context-free, and regular languages, respectively.

Notation 2.1. *To simplify notation when presenting the replacement rules of a grammar, we often write $\alpha \to \beta_1 \mid \beta_2 \mid \cdots \mid \beta_k$ to denote the k replacement rules $\alpha \to \beta_j$ for $j \in \{1, 2, \ldots, k\}$, where $\alpha, \beta_1, \beta_2, \ldots, \beta_k \in (V \cup \Sigma)^*$.*

The productions of a Type 2 grammar can be represented as a tree, for example, let $D_2 = (\Sigma, V, S, P)$ be the Type 2 grammar given by

$$\Sigma = \{a, b\},\ V = \{S\},\ P = \{S \to aSbS \mid \varepsilon\}.$$

This is a grammar for the *Dyck language*. The language corresponds to strings of matching open and closed brackets where a is an open bracket, and b is a close bracket. The word $aabaabbb$ is generated by the grammar D_2 and can be encoded with the *derivation tree* given in Figure 2.2. A derivation tree is a tree where the children of each node are ordered,

the word produced by the tree is obtained by performing an *in-order traversal* of its leaves. One can show that each word produced by the grammar D_2 will have exactly one derivation tree. A grammar is called *unambiguous* if it has exactly one derivation tree for each of its words. Moreover, if a language has an unambiguous grammar, then we will say that the language itself is *unambiguous*. Otherwise, if a language has no unambiguous grammar, then we say that it is *inherently ambiguous*.

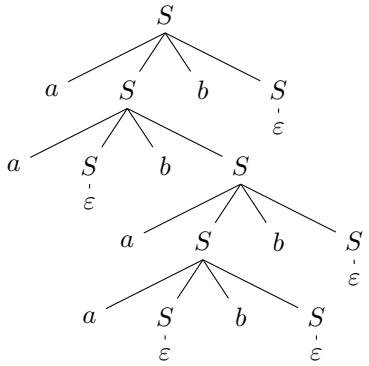

Figure 2.2: Derivation tree for *aabaabbb* in D_2

It is a result of Chomsky and Schützenberger that the generating function of any unambiguous context-free language is algebraic (see Lemma 2.8). This result allows us to show that certain languages are inherently ambiguous. For example, let L be the language

$$L = \{w \in \{a, \overline{a}, b, \overline{b}\}^* \mid |w|_a = |w|_{\overline{a}} \text{ or } |w|_b = |w|_{\overline{b}}\}.$$

From [48, p. 296] it is known that the generating function of L is

$$f(z) = \frac{2}{\pi} \int_0^{\pi/2} \frac{1}{\sqrt{1 - 16z^2 \sin^2 \theta}} \, \mathrm{d}\theta.$$

It can be shown that $f(z)$ is not algebraic and thus L is inherently ambiguous.

2.1.2. Finite-State Automata

The class of *finite-state automata* is equivalent in expressive power to the family of regular languages, that is, each regular language is recognised by some finite-state automaton, and the language recognised by each finite-state automaton is regular [92, Theorm 1.54].

A finite-state automaton is a finite directed edge-labelled graph with a distinguished initial vertex and a subset of the vertices labelled as *accepting*. A word is accepted by a finite-state automaton if there is a path from its initial vertex to an accepting vertex where the edge labels can be composed to form the word. Formally, we define a finite-state automaton as follows.

Definition 2.2. *A* finite-state automaton *is a 5-tuple* $M = (\Sigma, Q, A, q_0, \delta)$ *where*

- Σ *is the* input alphabet*;*
- Q *is a finite set of* states*;*
- $A \subseteq Q$ *is the set of* accepting states*;*
- $q_0 \in Q$ *is the* initial state*; and*
- $\delta \subseteq Q \times \Sigma \times Q$ *is a finite set known as the* transition relation*.*

The word $w = w_1 w_2 \cdots w_k \in \Sigma^*$ *is accepted by the finite-state automaton* M *if there is a finite sequence of states* $q_0, q_1, q_2, \ldots, q_k$ *with* $q_k \in A$ *and* $(q_i, w_{i+1}, q_{i+1}) \in \delta$ *for each* $i \in \{0, 1, 2, \ldots, k-1\}$*. The set of all such words is the language recognised by* M*. We can represent a finite-state automaton as a finite directed graph with vertex set* Q *where for each* $(p, \sigma, q) \in \delta$*, there is an edge from* p *to* q *labelled with* σ*.*

For example, the language

$$L = \{(ab)^n a^m \mid n, m \in \mathbb{N}_+\} \cup \{b^k \mid k \equiv 1 \pmod 3\}$$

is regular as it is recognised by $M = (\Sigma, Q, A, q_0, \delta)$ where

$$\begin{aligned} \Sigma &= \{a, b\},\ Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\},\ A = \{q_3, q_4\},\ \text{and} \\ \delta &= \{(q_0, a, q_1),\ (q_0, b, q_4),\ (q_1, b, q_2),\ (q_2, a, q_1),\ (q_2, a, q_3), \\ &\qquad (q_3, a, q_3),\ (q_4, b, q_5),\ (q_5, b, q_6),\ (q_6, b, q_4)\}. \end{aligned}$$

This finite-state automaton corresponds to the graph in Figure 2.3 where the double-circled nodes are accepting states.

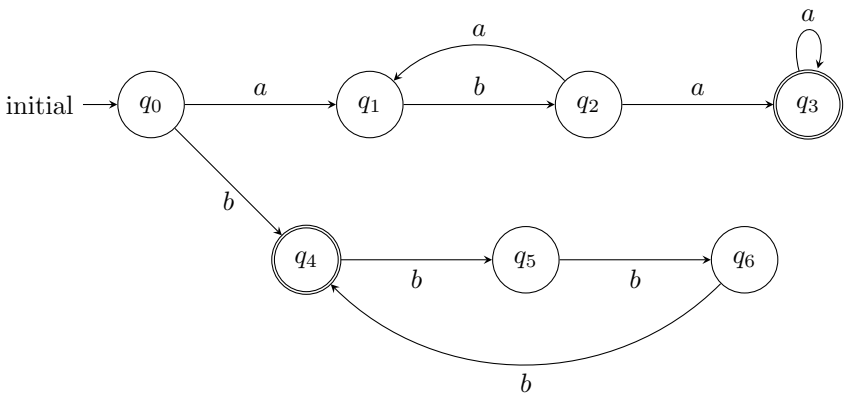

Figure 2.3: A finite-state automaton.

2.2. Generating Functions

In this section, we study the *univariate* and *multivariate generating functions* of formal languages. In particular, we study formal languages with *rational* and *holonomic generating functions*, as defined in Sections 2.2.1 and 2.2.3, respectively. Our aim in this

section is to provide enough background in analytic combinatorics so that we may prove Theorem A. For a more complete introduction, the reader is directed to [50].

The *univariate* generating function of a language is the power series whose coefficients count the number of words of a given length, that is, the power series defined as follows.

Definition 2.3. *The* univariate generating function *of a language $L \subseteq \Sigma^*$ is the formal power series $f(z) = \sum_{n=0}^{\infty} c_n z^n$ where each coefficient $c_n = \#\{w \in L \mid |w|_\Sigma = n\}$.*

The *multivarate generating function* of a language is the multivariate power series whose terms correspond to the number of occurrences of each letter in a word. To define these series, we first introduce the *Parikh map* as follows.

Definition 2.4. *Let $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$ be an ordered alphabet, then the* Parikh map *is the homomorphism $\Phi_\Sigma \colon \Sigma^* \to \mathbb{N}^m$ such that, for each word $w \in \Sigma^*$, we have*

$$\Phi_\Sigma(w) = (|w|_{\sigma_1}, |w|_{\sigma_2}, \ldots, |w|_{\sigma_m})$$

where $|w|_{\sigma_i}$ counts the number of occurrences of σ_i in w.

We may then define the multivariate generating function of a language as follows.

Definition 2.5. *The* multivariate generating function *of a language $L \subseteq \Sigma^*$ over an alphabet $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$ is the formal multivariate power series*

$$f(x_1, x_2, \ldots, x_m) = \sum_{(n_1, n_2, \ldots, n_m) \in \mathbb{N}^m} c(n_1, n_2, \ldots, n_m)\, x_1^{n_1} x_2^{n_2} \cdots x_m^{n_m}$$

where each coefficient

$$c(n_1, n_2, \ldots, n_m) = \#\{w \in L \mid \Phi_\Sigma(w) = (n_1, n_2, \ldots, n_m)\}$$

counts the number of words with a given Parikh image.

In the following subsections, we study classes of power series with increasing generality. In particular, we study the classes of *rational*, *algebraic*, and *holonomic* power series. For each of these classes of power series, we provide a family of formal languages such that the generating function of each language in the family lies within the class. In the following subsections, we write the notation $\mathbf{x} = (x_1, x_2, \ldots, x_m)$.

2.2.1. Rational Power Series

A (multivariate) power series $f(\mathbf{x})$ is *rational* if there are two polynomials $p(\mathbf{x})$ and $q(\mathbf{x})$, where $q(\mathbf{x})$ is nontrivial, such that $q(\mathbf{x})f(\mathbf{x}) = p(\mathbf{x})$. We denote this as $f(\mathbf{x}) = p(\mathbf{x})/q(\mathbf{x})$.

It is known that the (multivariate) generating function for a regular language is rational. For example, the regular language

$$L = \{a^i (ab)^j \mid i, j \in \mathbb{N}\}$$

has a multivariate generating function given by

$$\sum_{i=0}^{\infty}\sum_{j=0}^{\infty} x^i\,(xy)^j = \left(\sum_{i=0}^{\infty} x^i\right)\left(\sum_{j=0}^{\infty}(xy)^j\right) = \frac{1}{(1-x)(1-xy)}.$$

A useful property of univariate rational functions is that we may always find closed-form equations for their asymptotic behaviour, in particular, we have the following result.

Lemma 2.6 (Theorem IV.9 in [50]). *Suppose that $f(z) = \sum_{n=0}^{\infty} c_n z^n$ is rational with singularities at $\alpha_1, \alpha_2, \ldots, \alpha_k \in \mathbb{C}$. Then there are polynomials $p_1(n), \ldots, p_k(n) \in \mathbb{C}[z]$ such that for each sufficiently large n we have $c_n = \sum_{j=0}^{k} p_i(n)\alpha_j^{-n}$.*

Since the coefficients of univariate generating functions count words, we see that they have integer coefficients, and thus we may apply the Pólya-Carlson theorem as follows.

Lemma 2.7 (Carlson [22, p. 3]). *If $f(z)$ is a power series with integer coefficients that is complex analytic in the open unit disc, then $f(z)$ either has the unit circle as its natural boundary or is rational of the form $p(z)/(1-z^m)^n$ where $p(z) \in \mathbb{Z}[z]$ and $n, m \in \mathbb{N}$.*

From Lemma 2.7 we have the following characterisation of geodesic growth series with finitely many singularities. We make use of this corollary to prove Lemma 5.18 which we then use in the proof of Theorem A.

Corollary 2.7.1. *Let G be a group with a finite (weighted monoid) generating set S. If the geodesic growth series $f_S(z) = \sum_{n=0}^{\infty} \gamma_S(n) z^n$ has finitely many singularities, then G either has exponential geodesic growth with respect to S, or $f_S(z)$ is rational and G has polynomial geodesic growth of an integer degree with respect to S.*

Proof. From Lemma 1.3 we see that either the geodesic growth function $\gamma_S(n)$ has exponential growth or the geodesic growth rate $\alpha_S = \lim_{n\to\infty} \sqrt[n]{\gamma_S(n)} = 1$. In the latter case we apply Lemma 2.7 to show that $f_S(z)$ is a rational with singularities only at the m-th roots of unity for some $m \in \mathbb{N}$. From Lemma 2.6, we have

$$\gamma_S(n) = \sum_{j=1}^{k} p_j(n)\left(e^{2\pi i \cdot j/m}\right)^{-n} \tag{2.1}$$

for each sufficiently large n, were each $p_j(z)$ is a polynomial.

Since the geodesic growth function, $\gamma_S(n)$, is non-decreasing we have

$$\gamma_S(n) \leqslant \gamma_S(mn) = \sum_{j=1}^{k} p_j(mn) \quad \text{and} \quad \gamma_S(n) \geqslant \gamma_S(m\lfloor n/m \rfloor) = \sum_{j=1}^{k} p_j(m\lfloor n/m \rfloor) \tag{2.2}$$

for sufficiently large n.

Let d be the largest degree of any polynomial $p_i(z)$ in (2.1). Then, from the inequalities in (2.2) we see that there are positive constants $\alpha_1, \alpha_2 \in \mathbb{R}_{>0}$ such that

$$\alpha_1 n^d \leqslant \gamma_S(n) \leqslant \alpha_2 n^d$$

for every $n \in \mathbb{N}$. These bounds follow since $\gamma_S(0) = 1$ and $\gamma_S(n)$ is nondecreasing. □

2.2.2. Algebraic Power Series

A (multivariate) power series $f(\mathbf{x})$ is *algebraic* if there exists some nontrivial polynomial $a(\mathbf{x}, y) \in \mathbb{C}[\mathbf{x}, y]$ such that $f(\mathbf{x})$ satisfies the equation $a(\mathbf{x}, f(\mathbf{x})) = 0$. Notice that each rational function $f(\mathbf{x}) = p(\mathbf{x})/q(\mathbf{x})$ is also algebraic as can be seen from the choice of polynomial $a(\mathbf{x}, y) = q(\mathbf{x})y - p(\mathbf{x})$. We have the following formal language characterisation.

Lemma 2.8 (Chomsky and Schützenberger [26])**.** *The (multivariate) generating function of an unambiguous context-free language is algebraic.*

For example, the context-free language

$$L = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$$

has an algebraic multivariate generating function

$$f(x, y) = \sum_{k=0}^{\infty} \binom{2k}{k} x^k y^k = \frac{1}{\sqrt{1 - 4xy}}.$$

This equality is given in [95, Eq. (2.5.11) on p. 53].

2.2.3. Holonomic Power Series

In this subsection we provide a background to the class of *holonomic* power series. Some authors use the term *D-finite* or *differentiably finite* to refer to the class of single-variable holonomic functions, or as a synonym for holonomic. This class of power series is interesting due to its closure properties, and that the coefficients of a univariate holonomic power series are *easy* to compute, i.e., the sequence of coefficients satisfies a recurrence relation with polynomial coefficients known as a *P-recurrence* [50, p. 748]. In Section 2.3.1 we study the family of *linearly constrained languages* which have holonomic (multivariate) generating functions. We extend this result in Chapter 4 to the family of *polyhedrally constrained languages*. In Theorem A, we use polyhedrally constrained languages to show that the geodesic growth series of each virtually abelian group is holonomic.

A (multivariate) generating function $f(\mathbf{x})$ is *holonomic* if the span of

$$X_f = \left\{ \partial_{x_1}^{k_1} \partial_{x_2}^{k_2} \cdots \partial_{x_m}^{k_m} f(\mathbf{x}) \;\middle|\; k_1, k_2, \ldots, k_m \in \mathbb{N} \right\}$$

over $\mathbb{C}(\mathbf{x})$ is a finite-dimensional vector space $V_f \subseteq \mathbb{C}((\mathbf{x}))$, or equivalently, $f(\mathbf{x})$ is holonomic if it is a solution to a system of differential equations of the form

$$\begin{aligned}
\partial_{x_1}^{k_1+1} f(\mathbf{x}) + r_{1,k_1}(\mathbf{x})\, \partial_{x_1}^{k_1} f(\mathbf{x}) + \cdots + r_{1,1}(\mathbf{x})\, \partial_{x_1} f(\mathbf{x}) + r_{1,0}(\mathbf{x}) f(\mathbf{x}) &= 0 \\
\partial_{x_2}^{k_2+1} f(\mathbf{x}) + r_{2,k_2}(\mathbf{x})\, \partial_{x_2}^{k_2} f(\mathbf{x}) + \cdots + r_{2,1}(\mathbf{x})\, \partial_{x_2} f(\mathbf{x}) + r_{2,0}(\mathbf{x}) f(\mathbf{x}) &= 0 \\
&\vdots \\
\partial_{x_m}^{k_m+1} f(\mathbf{x}) + r_{m,k_m}(\mathbf{x})\, \partial_{x_m}^{k_m} f(\mathbf{x}) + \cdots + r_{m,1}(\mathbf{x})\, \partial_{x_m} f(\mathbf{x}) + r_{m,0}(\mathbf{x}) f(\mathbf{x}) &= 0
\end{aligned}$$

where each $r_{i,j}(\mathbf{x}) \in \mathbb{C}(\mathbf{x})$ is a rational function. Notice here that there is one differential equation for each independent variable x_i. This equivalent definition is the reason that some authors prefer the name D-finite and differentiably finite.

The class of holonomic power series satisfy many nice closure properties, however, in this thesis we only require the following.

Lemma 2.9 (Proposition 2.3 in [73])**.** *The class of holonomic power series over the variables* $\mathbf{x}$ *is closed under addition and multiplication. If* $f(\mathbf{x})$ *is a holonomic power series with* $\mathbf{x} = (x_1, x_2, \ldots, x_m)$*, and* $a_1(\mathbf{y}), a_2(\mathbf{y}), \ldots, a_m(\mathbf{y})$ *are algebraic power series, then* $g(\mathbf{y}) = f(a_1(\mathbf{y}), a_2(\mathbf{y}), \ldots, a_m(\mathbf{y}))$ *is also holonomic if it is defined. Each algebraic power series (and thus each rational function) is holonomic.*

For univariate holonomic power series with integer coefficients, we may also apply Pólya-Carlson theorem, as given in Lemma 2.7. That is, holonomic functions may have only finitely many singularities as in the following lemma.

Lemma 2.10 (Theorem 1 in [49])**.** *A univariate holonomic function may only have finitely many singularities.*

2.3. Constrained Languages

Linearly constrained languages, defined below in Section 2.3.1, were introduced by Massazza [76] as an example of a family of languages with holonomic univariate generating functions. In this section, we define the more general class of *constrained languages*, and show that Massazza's result holds for multivariable generating functions. In Chapter 4 we provide a new generalisation of linearly constrained languages, known as *polyhedrally constrained*, which we use in the proof of Theorem A.

Definition 2.11. *Let* $U \subseteq \Sigma^*$ *be an unambiguous context-free language and let* $\mathcal{C} \subseteq \mathbb{Z}^{|\Sigma|}$*, then* $L(U, \mathcal{C}) = \{w \in U \mid \Phi_\Sigma(w) \in \mathcal{C}\}$ *is a* constrained language *(where* $\Phi_\Sigma \colon \Sigma^* \to \mathbb{N}^{|\Sigma|}$ *is the Parikh map as given in Definition 2.4).*

We then study families of constrained languages by placing restrictions on the sets $\mathcal{C} \subseteq \mathbb{Z}^n$. Informally, the family of *linearly constrained languages* results from requiring that $\mathcal{C}$ corresponds to the integer solutions of a system of linear equations.

2.3.1. Linear Constraints

Modifying the notation of Massazza [76], we define *n-atoms* and *n-constraints* as follows.

Definition 2.12. *A subset of* $\mathbb{Z}^n$ *is an* n-atom *if it can be expressed as* $\{v \in \mathbb{Z}^n \mid a \cdot v = b\}$ *or* $\{v \in \mathbb{Z}^n \mid a \cdot v > b\}$ *where* $a \in \mathbb{Z}^n$ *and* $b \in \mathbb{Z}$*. An* n-constraint *is a Boolean expression of* n*-atoms, that is, a finite expression of* n*-atoms using intersection, union, and complement with respect to* $\mathbb{Z}^n$*.*

For example,

$$\{(x, y) \in \mathbb{Z}^2 \mid \text{either } x = 1 \text{ and } y > 10, \text{ or } x \neq 1 \text{ and } 2x - 3y > 4\}$$

is a 2-constraint as it can be written as the Boolean expression

$$\{v \in \mathbb{Z}^2 \mid (1, 0) \cdot v = 1\} \cap \{v \in \mathbb{Z}^2 \mid (0, 1) \cdot v > 10\} \\ \cup \left(\mathbb{Z}^2 \setminus \{v \in \mathbb{Z}^2 \mid (1, 0) \cdot v = 1\}\right) \cap \{v \in \mathbb{Z}^2 \mid (2, -3) \cdot v > 4\}.$$

Massazza defined the family of *linearly constrained languages* as follows.

Definition 2.13. *If $\mathcal{C}$ is an n-constraint, then $L(U, \mathcal{C})$ is a* linearly constrained language.

Massazza [76] introduced linearly constrained languages as a family of languages with holonomic univariate generating functions. Massazza proved this by first showing that linearly constrained languages have holonomic multivariate generating functions. Thus, we have the more general result given in Proposition 2.14.

Proposition 2.14. *The multivariate generating function of a linearly constrained language is holonomic.*

Proof. See the proof of Theorem 2 in [76]. □

For example, let $L = L(U, \mathcal{B})$ be the linearly constrained language with

$$U = \{a, b, c\}^* \quad \text{and} \quad \mathcal{B} = \{(n, n, n) \in \mathbb{Z}^3 \mid n \in \mathbb{N}\}.$$

It was shown in [76, Example 2] that L has a multivariate generating function of

$$f(x, y, z) = \sum_{n \in \mathbb{N}} \frac{(3n)!}{(n!)^3} x^n y^n z^n.$$

From Proposition A.2 (see the appendix), we see that $f(x, y, z)$ satisfies the system of differential equations

$$(x^2 - 27x^3yz)\partial_x^2 f(x, y, z) + (x - 54x^2yz)\partial_x f(x, y, z) - 6xyz\, f(x, y, z) = 0$$
$$(y^2 - 27xy^3z)\partial_y^2 f(x, y, z) + (y - 54xy^2z)\partial_y f(x, y, z) - 6xyz\, f(x, y, z) = 0$$
$$(z^2 - 27xyz^3)\partial_z^2 f(x, y, z) + (z - 54xyz^2)\partial_z f(x, y, z) - 6xyz\, f(x, y, z) = 0.$$

Hence, the generating function $f(x, y, z)$ is holonomic.

2.4. Blind Multicounter Automata

The class of *blind multicounter automata* was introduced by Greibach [52], where they were shown [52, Theorem 2] to be equivalent in expressive power to the class of *reversal-bounded multicounter automata* as introduced by Baker and Book [8]. Moreover, it was shown in

[65, § 2.2] that the class of reversal-bounded multicounter automata, and thus the class of *blind multicounter automata*, are equivalent in expressive power to the class of *Parikh automata* introduced by Klaedtke and Rueß [64].

We say that a language is *blind multicounter* if it is accepted by a *blind multicounter automata*. It was shown by Elder, Kambites and Ostheimer [42] that the word problem of a group is blind multicounter if and only if the group is virtually abelian. In Theorem C we show that the language of geodesics for a virtually abelian group is blind multicounter.

Informally, a *blind k-counter automaton* is a nondeterministic finite-state acceptor with a one-way input tape and k integer counters. The machine is allowed to increment and decrement its counters by fixed amounts during transitions where each transition does not depend on the state of the counters. A computation of a blind k-counter automata begins with zero on all its counters, and accepts when it is in an accepting state with all input consumed and zero on each counter. A language L is called *blind multicounter* if it is accepted by a blind k-counter automaton for some $k \in \mathbb{N}$. The family of blind multicounter languages satisfies the hierarchy given in Figure 2.4. We prove the correctness of this diagram at the end of this section, that is, after we provide the formal definition of blind multicounter language in Definition 2.16.

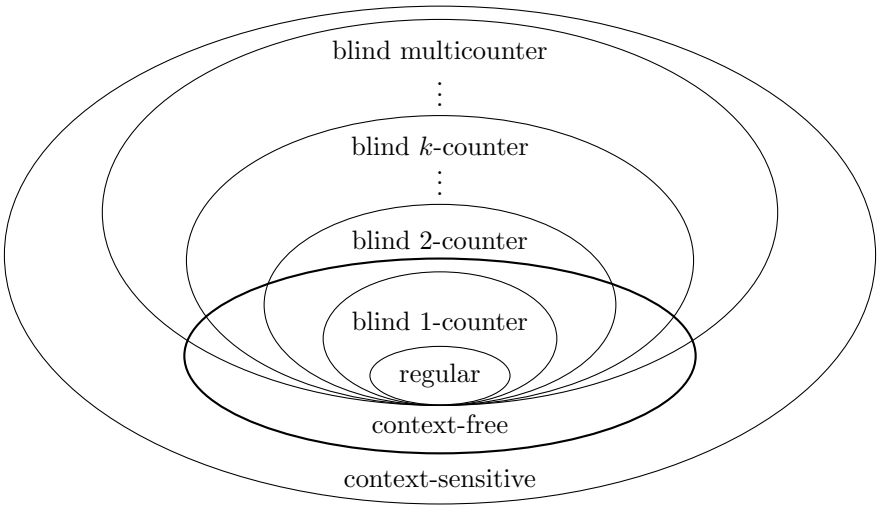

Figure 2.4: Hierarchy of blind multicounter language.

Our definition of a blind multicounter automaton differs slightly from the one given by Greibach [52]. In particular, we introduce $\mathfrak{e}$ as an end of input symbol, and allow our automata to add and subtract any constant vector from their counters on a transition instead of only allowing basis vectors. It is clear that this does not increase the expressive power of our model. Formally, we define a blind k-counter automaton as follows.

Definition 2.15. *Let $k \in \mathbb{N}$, then a* blind k-counter automaton *is a* 6*-tuple of the form $M = (Q, \Sigma, \delta, q_0, F, \mathfrak{e})$ where*

1. *Q is a finite set of* states*;*
2. *Σ is a finite* input alphabet*;*

3. *δ is a finite subset of*
$$\big(Q \times (\Sigma \cup \{\varepsilon, \mathfrak{e}\})\big) \times \big(Q \times \mathbb{Z}^k\big),$$
where ε is the empty word, which we call the transition relation*;*
4. *$q_0 \in Q$ is the* initial state*;*
5. *$F \subseteq Q$ is the set of* final states*; and*
6. *$\mathfrak{e} \notin \Sigma$ is the* end of tape symbol*.*

Let $M = (Q, \Sigma, \delta, q_0, F, \mathfrak{e})$ be a blind k-counter automaton. Then M begins in state q_0 with zero on all its counters. Suppose that there is a transition relation $((q, a), (p, v)) \in \delta$ with $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon, \mathfrak{e}\}$ and $v \in \mathbb{Z}^k$; if M is in state q with a as the next letter on its input tape, then it can transition to state p after adding v to its counters and consuming a from its input tape. The machine accepts if it is in an accepting state $q \in F$, has no letters remaining on its input tape, and has zero on all its counters.

Formally, we represent the configuration of a blind k-counter automaton M as an *instantaneous description* of the form

$$(q, (c_1, c_2, \ldots, c_k), \sigma\mathfrak{e}) \in Q \times \mathbb{Z}^k \times \Sigma^*\mathfrak{e}$$

where $q \in Q$ is the *current state*, $(c_1, c_2, \ldots, c_k) \in \mathbb{Z}^k$ are the values of the counters, and $\sigma \in \Sigma^*$ is the sequence of letters which have yet to be consumed. Let C_1 and C_2 be instantaneous descriptions for the configuration of M. Then we write $C_1 \vdash C_2$ if M can move from configuration C_1 to C_2 in a single transition. Formally, we interpret the transition relation δ as follows.

For each transition relation of the form $((q, s), (p, v)) \in \delta$ with $s \in \Sigma \cup \{\varepsilon\}$, and for each $\sigma = s\sigma' \in \Sigma^*$, we have transitions of the form

$$(q, (c_1, c_2, \ldots, c_k), \sigma\mathfrak{e}) \vdash (p, (c_1 + v_1, c_2 + v_2, \ldots, c_k + v_k), \sigma'\mathfrak{e}).$$

Moreover, for each relation $((q, \mathfrak{e}), (p, v)) \in \delta$ we have transitions

$$(q, (c_1, c_2, \ldots, c_k), \mathfrak{e}) \vdash (p, (c_1 + v_1, c_2 + v_2, \ldots, c_k + v_k), \mathfrak{e}).$$

Notice that we do not consume the end of tape symbol $\mathfrak{e}$.

We then write $\vdash^*$ for the transitive symmetric closure of $\vdash$, that is, we have $C_1 \vdash^* C_2$ if M can move from configuration C_1 to C_2 within finitely many transitions. We say that a word $\sigma \in \Sigma^*$ is accepted by M if

$$(q_0, (0, 0, \ldots, 0), \sigma\mathfrak{e}) \vdash^* (q, (0, 0, \ldots, 0), \mathfrak{e})$$

for some $q \in F$. We define the language accepted by a blind multicounter automaton as follows.

Definition 2.16. *Let M be a blind multicounter automaton. Then*

$$L(M) = \{\sigma \in \Sigma^* \mid (q_0, (0, 0, \ldots, 0), \sigma\mathfrak{e}) \vdash^* (q, (0, 0, \ldots, 0), \mathfrak{e}) \text{ where } q \in F\}$$

is the blind multicounter language *accepted by M.*

The family of blind multicounter languages satisfy the language hierarchy in Figure 2.4, in particular, we may construct a language for each region of the diagram in Figure 2.4 as follows. We see that the class of finite-state automata is equivalent to the class of blind 0-counter automata, and that each blind k-counter language is also a blind $(k+1)$-counter language. We see that blind 1-counter languages form a subfamily of the context-free languages. Moreover, from [52, Theorem 1] it can be seen that the class of blind multicounter languages is a subclass of context-sensitive languages. From [89] it is known that $F_2 \times F_2$ has a context-sensitive word problem. It is a classic result by Muller and Schupp [78] that the word problem for a group is context-free if and only if the group is virtually free. Moreover, it was shown in [42] that the word problem for a group is blind k-counter if and only if the group is virtually $\mathbb{Z}^m$ for some $m \leqslant k$. From these characterisations we see that the word problem for the free group F_2 is context-free but not blind multicounter; for each $k \geqslant 2$, the word problem for $\mathbb{Z}^k$ is blind k-counter but not context-free; the word problem for $\mathbb{Z}^{k+1}$ is blind $(k+1)$-counter but not blind k-counter; and that the word problem for $F_2 \times F_2$ is context-sensitive and neither context-free nor blind multicounter. From the proof of Theorem 5 in [52], we see that

$$L_k = \{a_1^{n_1} a_2^{n_2} \cdots a_k^{n_k} b_k^{n_k} \cdots b_2^{n_2} b_1^{n_1} \mid n_1, n_2, \ldots, n_k \in \mathbb{N}\}$$

is context-free and blind k-counter, but not blind $(k-1)$-counter.

2.5. ET0L Languages

The family of *Extended Tabled 0-interaction Lindenmayer* (*ET0L*) languages and their deterministic counterpart *EDT0L* were introduced and studied by Rozenberg [87]. The class of ET0L language results from modifying the grammar of a context-free language, in particular, we demand that a replacement is made for each non-terminal letter at the same time in the sense of an *L-system*. We provide a formal definition of this family of languages in Definition 2.17.

In recent publications, the family of *ET0L* languages and their deterministic counterpart, *EDT0L*, have found their place as a natural choice for modelling group-theoretic problems. In particular, it is known that the solutions to equations over free monoids with involution [37], hyperbolic groups [30], virtually abelian groups [45], and right-angled Artin groups [38] can be expressed as EDT0L languages, and that these languages are effectively constructable. Moreover, it was shown by Ciobanu, Elder and Ferov [32] that many of the existing problems in group theory that were known to be context-sensitive (or indexed), are in-fact ET0L.

It was shown by Holt and Röver [63] that the co-word problem for *bounded automata groups* is indexed. Ciobanu, Elder and Ferov [32] strengthened this result for the case of Grigorchuk group, in particular, they provided an explicit ET0L grammar for the co-word problem of Grigorchuk's group. In Theorem C, we complete this project by showing that all bounded automata groups have ET0L co-word problems. We accomplish this with the use of an equivalent machine model known as a *cspd automaton* (see Section 2.5.1). We provide a self-contained proof of this equivalence in Section 2.5.2.

In the definition of formal grammar we gave in Section 2.1, we were allowed to apply the grammar rules in any order to any part of a word. One generalisation of this is *Lindenmayer-systems* as introduced by Lindenmayer [72] to model the growth of filamentous organisms, e.g., algae. In this model, we demand that a grammar rule be applied to each nonterminal in parallel. Informally, ET0L languages are Lindenmayer-systems with context-free replacement rules which are collected into *tables*. Formally, we define ET0L languages as follows.

A *table*, τ, is a finite set of context-free replacement rules where each non-terminal, $X \in V$, has at least one replacement in τ. For example, let $\Sigma = \{a, b\}$ and $V = \{S, A, B\}$, then the following are tables.

$$\alpha\colon \begin{cases} S \to SS \mid S \mid AB \\ A \to A \\ B \to B \end{cases} \qquad \beta\colon \begin{cases} S \to S \\ A \to aA \\ B \to bB \end{cases} \qquad \gamma\colon \begin{cases} S \to S \\ A \to \varepsilon \\ B \to \varepsilon \end{cases} \tag{2.3}$$

We apply a table, τ, to the word $w \in (\Sigma \cup V)^*$ to obtain a word w', written $w \to^\tau w'$, by performing a replacement in τ to each non-terminal in w. If a table includes more than one rule for some non-terminal, we nondeterministically and independently apply a replacement to each occurrence. For example, with $w = SSSS$ and α as in (2.3), we can apply α to w to obtain $w' = SABSSAB$. Given a sequence of tables τ_1, τ_2, …, τ_k, we will write $w \to^{\tau_1\tau_2\cdots\tau_k} w'$ if there is a sequence of words $w = w_1, w_2, \ldots, w_{k+1} = w'$ such that $w_j \to^{\tau_j} w_{j+1}$ for each j. Notice here that the tables are applied from left to right.

Definition 2.17 (Asveld [5])**.** *An* ET0L grammar *is a 5-tuple* $G = (\Sigma, V, T, \mathcal{R}, S)$*, 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.

The ET0L language *produced by the grammar* G*, denoted* $L(G)$*, is*

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

Moreover, G *is an* EDT0L grammar*, and* $L(G)$ *an* EDT0L language*, if for each table* τ_i *has exactly one replacement for each non-terminal letter.*

For example, let α, β and γ as in (2.3), then the language produced 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 [92, Theorem 2.34]) that this language is not context-free. We see that each context-free language is ET0L, in particular, for each context-free grammar (Σ, V, S, P) we may construct an ET0L grammar $(\Sigma, V, T, \mathcal{R}, S)$ with $\mathcal{R} = T^*$ where T contains one table $\tau = P \cup \{v \mapsto v \mid v \in V\}$. Thus, the family of ET0L languages contain the context-free languages as a proper sub-family.

2.5.1. CSPD Automata

An alternative method to show that a language is ET0L is by making use of an equivalent machine model known as a *cspd automaton*. In Section 2.5.2, we show that ET0L languages and cspd automata are equivalent in expressive power.

A *cspd automaton*, studied in [69], is a nondeterministic finite-state automaton with a one-way input tape, and access to both a *check-stack* (with stack alphabet Δ) and a *pushdown stack* (with stack alphabet Γ), where access to these two stacks is tied in a very particular way. The execution of a cspd machine is separated into two stages.

In the first stage the machine is allowed to push letters onto its check-stack but *not* its pushdown, and further, the machine will not be allowed to read from its input tape. Thus, the set of all possible check-stacks that can be constructed in this stage forms a regular language which we will denote as $\mathcal{R}$.

In the second stage, the machine can no longer alter its check-stack, but is allowed to access its pushdown and input tape. We restrict the machine's access to its stacks so that it can only move along its check-stack by pushing and popping items to and from its pushdown. In particular, every time the machine pushes a value onto the pushdown it will move up the check-stack, and every time it pops a value off of the pushdown it will move down the check-stack. See Figure 2.5 for an example of this behaviour.

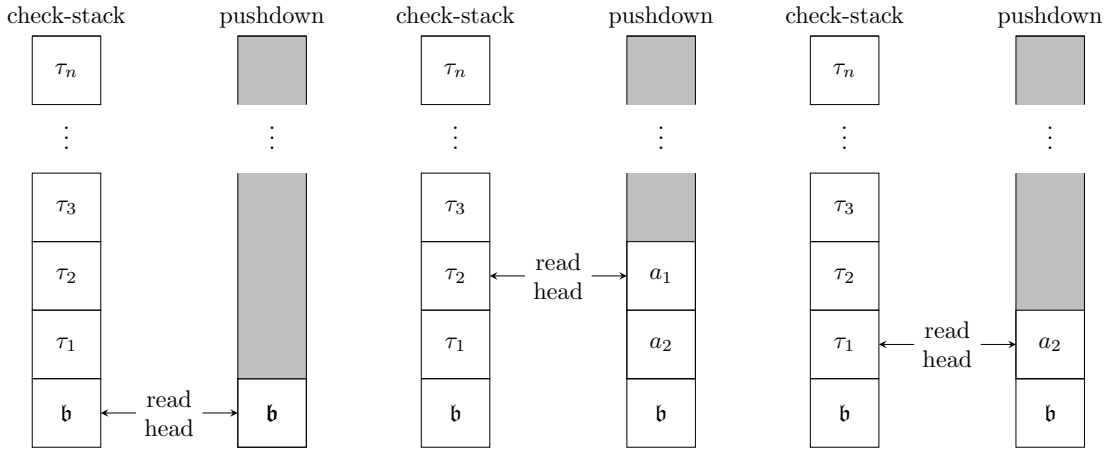

Figure 2.5: An example of a cspd machine pushing $w = a_1a_2$, where $a_1, a_2 \in \Delta$, onto its pushdown stack, then popping a_1 (as read from left to right).

We define a cspd machine formally as follows.

Definition 2.18. *A* cspd machine *is a 9-tuple* $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, \mathfrak{b}, \mathcal{R}, \theta, q_0, F)$*, where*

1. Q *is the set of* states*;*
2. Σ *is the* input alphabet*;*
3. Γ *is the* alphabet for the pushdown*;*
4. Δ *is the* alphabet for the check-stack*;*
5. $\mathfrak{b} \notin \Delta \cup \Gamma$ *is the* bottom of stack symbol*;*
6. $\mathcal{R} \subseteq \Delta^*$ *is a* regular language of allowed check-stacks*;*

7. θ *is a finite subset of*

$$(Q \times (\Sigma \cup \{\varepsilon\}) \times ((\Delta \times \Gamma) \cup \{(\varepsilon, \varepsilon), (\mathfrak{b}, \mathfrak{b})\})) \times (Q \times (\Gamma \cup \{\mathfrak{b}\})^*),$$

called the transition relation *(see below for allowable elements of θ);*
8. $q_0 \in Q$ *is the* start state*; and*
9. $F \subseteq Q$ *is the set of* accepting states.

In its initial configuration, the machine is in state q_0, the check-stack will contain $\mathfrak{b}w$ for some nondeterministic choice of $w \in \mathcal{R}$, the pushdown will contain only the letter $\mathfrak{b}$, the read-head for the input tape will be at its first letter, and the read-head for the machine's stacks will be pointing to the $\mathfrak{b}$ on both stacks. From here, the machine will follow transitions as specified by θ. Each such transition will have one of the following three forms, where $a \in \Sigma \cup \{\varepsilon\}$, $p, q \in Q$ and $w \in \Gamma^*$.

1. $((p, a, (\mathfrak{b}, \mathfrak{b})), (q, w\mathfrak{b})) \in \theta$ meaning that if the machine is in state p, sees $\mathfrak{b}$ on both stacks and is able to consume a from its input; then it can follow this transition to consume a, push w onto the pushdown and move to state q.
2. $((p, a, (d, g)), (q, w)) \in \theta$ where $(d, g) \in \Delta \times \Gamma$, meaning that if the machine is in state p, sees d on its check-stack, g on its pushdown, and is able to consume a from its input; then it can follow this transition to consume a, pop g, then push w and move to state q.
3. $((p, a, (\varepsilon, \varepsilon)), (q, w)) \in \theta$ meaning that if the machine is in state p and can consume a from its input; then it can follow this transition to consume a, push w and move to state q.

In the previous three cases, $a = \varepsilon$ corresponds to a transition in which the machine does not consume a letter from input. We use the convention that, if $w = w_1 w_2 \cdots w_k$ with each $w_j \in \Gamma$, then the machine will push w_k first, followed by the w_{k-1} and so forth. The machine accepts if it has consumed all its input and is in an accepting state $q \in F$.

In [69] van Leeuwen proved that the family of languages recognised by cspd automata is precisely the class of ET0L languages. For completeness, in the following subsection we provide a self-contained proof of this equivalence.

2.5.2. Equivalence of CSPD and ET0L

We now provide our own self-contained proof of the equivalence between the family of ET0L languages and the class of languages recognised by cspd automata. We begin by introducing some additional notation and a normal form for cspd automata, then we prove our result in Lemmas 2.22 and 2.23.

Additional Notation

We define a non-terminal $\mathfrak{d}$ which we call a *dead-end symbol*. If an ET0L grammar has the dead-end symbol, then we demand that $\mathfrak{d}$ is not a terminal and that each table can only map $\mathfrak{d}$ to itself, i.e., $\mathfrak{d} \to \mathfrak{d}$. Thus, if a table induces a letter $\mathfrak{d}$, then there is no way to remove it to generate a word in the associated language.

For simplicity when presenting tables, if a replacement is not specified for a particular variable X, then it should be assumed that the replacement rule $X \to X$ is in the table.

We introduce the following generalisation of ET0L grammars. We use this generalisation to construct an ET0L grammar from a cspd automaton in Lemma 2.23.

Lemma 2.19 (Christensen [28])**.** *The class of ET0L grammars does not gain any expressive power if each replacement rule has the form $\tau : X \to L_{X,\tau}$ where each $L_{X,\tau}$ is an ET0L language, that is, if we allow τ to replace instances of the variable X with nondeterministic choices of words from $L_{X,\tau}$.*

Proof. Let $G = (\Sigma, V, T, \mathcal{R}, S)$ be a grammar in this extended form, that is, where each replacement rule maps X into any word in some ET0L language $L_{X,\tau}$. Assume without loss of generality that every terminal is also a non-terminal, i.e., $\Sigma \subseteq V$ (this is done by first adding replacement rules $\tau\colon a \to a$ for each table $\tau \in T$ and each $a \in \Sigma \setminus V$; then we add the letters of Σ to V. It is clear that this modified grammar generates the same language).

For each language $L_{X,\tau}$ in the grammar G, let

$$G_{X,\tau} = (\Sigma_{X,\tau}, V_{X,\tau}, T_{X,\tau}, \mathcal{R}_{X,\tau}, S_{X,\tau})$$

be an ET0L grammar such that $L_{X,\tau} = L(G_{X,\tau})$. Notice here that $\Sigma_{X,\tau}$ must be a subset of V such that the language $L_{X,\tau}$ generates words in V^*.

For each $X \in V$, we define two disjoint copies denoted as $X^{(1)}$ and $X^{(2)}$; and we demand that these copies are disjoint to letters in the alphabets $V_{Y,\tau}$ and $\Sigma_{Y,\tau}$ for each $Y \in V$ and $\tau \in T$.

We define two ET0L tables α and κ such that, for each $X \in V$, we have replacement rules $\alpha\colon X \to X^{(1)}$ and $\kappa\colon X^{(1)} \to \mathfrak{d}$.

For each table $\tau \in T$ and non-terminal $X \in V$, we define ET0L tables

$$\beta_X^\tau\colon X^{(1)} \to X^{(1)} \mid S_{X,\tau} \quad \text{and} \quad \gamma_X^\tau\colon \begin{cases} Y \to Y^{(2)} & \text{for } Y \in \Sigma_{X,\tau} \subseteq V, \\ Z \to \mathfrak{d} & \text{for } Z \in V_{X,\tau} \setminus \Sigma_{X,\tau}. \end{cases}$$

Given a $\tau \in T$, it can be seen that τ is equivalent to the regular expression

$$\tau' = \alpha \left(\beta_{X_1}^\tau \mathcal{R}_{X_1,\tau} \gamma_{X_1}^\tau\right)^* \left(\beta_{X_2}^\tau \mathcal{R}_{X_2,\tau} \gamma_{X_2}^\tau\right)^* \cdots \left(\beta_{X_k}^\tau \mathcal{R}_{X_k,\tau} \gamma_{X_k}^\tau\right)^* \kappa$$

where $\{X_1, X_2, \ldots, X_k\} = V$. Thus, after replacing each τ in a regular expression for $\mathcal{R}$ with its corresponding expression τ', we obtain a regular language which we denote $\mathcal{R}'$. Thus, it can be seen that the grammar G is equivalent to an ET0L grammar with rational control given by $\mathcal{R}'$. □

In the proof of Lemma 2.23, we begin by normalising a given cspd automaton as follows.

Lemma 2.20. *Given a cspd automaton, $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, \mathfrak{b}, \mathcal{R}, \theta, q_0, F)$, we may assume without loss of generality that:*

1. *the pushdown is never higher than the check-stack;*

2. *there is only one accepting state, i.e.,* $\{q_{\text{accept}}\} = F$*;*
3. *the pushdown is empty when* $\mathcal{M}$ *enters the accepting state* q_{accept}*;*
4. *transitions to the accepting state* q_{accept} *do not modify the pushdown;*
5. *each transition to states other than* q_{accept} *either pushes exactly one letter to the pushdown or pops exactly one letter from the pushdown.*

Proof. Let $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, \mathfrak{b}, \mathcal{R}, \theta, q_0, F)$ be a cspd machine.

For assumption 1, let $N \in \mathbb{N}$ be an upper bound on the number of letters any transition of $\mathcal{M}$ can push. That is, N is such that, given any transition $((q, a, (d, g)), (p, b_1 b_2 \cdots b_k)) \in \theta$ where each $b_j \in \Gamma$, it is the case that $k \leqslant N$. We now add a disjoint letter $\mathfrak{t}$ to the check-stack alphabet Δ. Thus, $\mathcal{M}$ has no available transitions when it sees $\mathfrak{t}$ on its check-stack. We thus satisfy assumption 1 after replacing the regular language of check-stacks with $\mathcal{R}\mathfrak{t}^N$ (where $\mathfrak{t}^N$ is a sequence of N letters $\mathfrak{t}$'s).

For assumptions 2–4 we introduce states q_{finish} and q_{accept} disjoint from all other states in Q. For each $(d, g) \in \Delta \times \Gamma$ and each $q \in F$ we add

$$((q, \varepsilon, (d, g)), (q_{\text{finish}}, \varepsilon)),\ ((q_{\text{finish}}, \varepsilon, (d, g)), (q_{\text{finish}}, \varepsilon))$$

so that we can empty the pushdown after reaching an accepting state $q \in F$. Further, for each state $q \in F$, we add transitions

$$((q, \varepsilon, (\mathfrak{b}, \mathfrak{b})), (q_{\text{accept}}, \mathfrak{b})),\ ((q_{\text{finish}}, \varepsilon, (\mathfrak{b}, \mathfrak{b})), (q_{\text{accept}}, \mathfrak{b}))$$

so that we can move to state q_{accept} once the pushdown has been emptied.

Thus, we now replace the set of accepting states, F, with $\{q_{\text{accept}}\}$ to obtain an equivalent machine that satisfies assumptions 2–4.

For assumption 5, we only need to consider transitions of the form

$$((p, \alpha, (d, g)), (q, a_1 a_2 \cdots a_k)) \tag{2.4}$$

where $p, q \in Q$ with $q \neq q_{\text{accept}}$, $(d, g) \in \Delta \times \Gamma \cup \{(\mathfrak{b}, \mathfrak{b})\}$, $\alpha \in \Sigma^*$, and each $a_j \in \Gamma \cup \{\mathfrak{b}\}$ with either $a_k \neq g$ or $k > 2$.

Given a transition as in (2.4), we add states $p^q_{a_1}$, $p^q_{a_1 a_2}$, $\ldots$, $p^q_{a_1 a_2 \cdots a_k}$; and for each $(b, c) \in \Delta \times \Gamma \cup \{(\mathfrak{b}, \mathfrak{b})\}$ and $j \in \mathbb{N}$ with $1 < j \leqslant k$, we add transitions

$$((p^q_{a_1 a_2 \cdots a_j}, \varepsilon, (b, c)), (p^q_{a_1 a_2 \cdots a_{j-1}}, a_j c)),\ ((p^q_{a_1}, \varepsilon, (b, c)), (q, a_1 c)),$$

so that, from state $p^q_{a_1 a_2 \cdots q_j}$, we go through a sequence of transitions which push the word $a_1 a_2 \cdots a_j$ and end in the state q. Moreover, if $a_k \neq g$ in our given transition, then we add a transition

$$((p, \alpha, (d, g)), (p^q_{a_1 a_2 \cdots a_k}, \varepsilon)),$$

otherwise we add a transition

$$((p, \alpha, (d, g)), (p^q_{a_1 a_2 \cdots a_{k-2}}, a_{k-1} g)).$$

Notice that with the addition of these states and transitions, we can remove all transitions as in (2.4) and still recognise the same language.

This completes the proof. □

Proof of Equivalence

In this section we prove the following equivalence.

Proposition 2.21 (van Leeuwen [69])**.** *The class of ET0L languages is equivalent to the class of languages recognised by cspd automata.*

We now provide our own self-contained proof of the equivalence given in Proposition 2.21. The proof is divided into Lemmas 2.22 and 2.23. We begin as follows by showing that ET0L languages form a subfamily of languages recognised by cspd automata.

Lemma 2.22. *For each ET0L language there is an equivalent cspd automaton.*

Proof. Let L be a given ET0L language with grammar $G = (\Sigma, V, T, \mathcal{R}, S)$, i.e., $L = L(G)$. Thus, in this proof we construct a cspd machine $\mathcal{M}$ which accepts precisely the language L by emulating derivations of words with respect to G.

Let $w \in L$. Then, by the definition of an ET0L grammar, there exists a sequence of tables $\alpha = \alpha_1\alpha_2 \cdots a_k \in \mathcal{R}$ for which $S \to^{\alpha} w$. Thus, the idea of this construction is that $\mathcal{M}$ accepts the word w by choosing its check-stack to represent such a sequence α, then $\mathcal{M}$ emulates a derivation of w from S with the use of its pushdown and reference to the check-stack.

We now give a description of this construction. We begin by choosing the alphabets and states for $\mathcal{M}$ as follows.

The input alphabet of $\mathcal{M}$ is given by Σ. The alphabet of the check-stack is given by $\Delta = T \cup \{\mathfrak{t}\}$ where $\mathfrak{t}$ is used to denote the top of the check-stack. Further, the regular language of allowed check-stacks is given by $\mathcal{R}\mathfrak{t}$ where $\mathcal{R}$ is the rational control of the grammar G.

The alphabet for the pushdown, Γ, will include letters $[\![S]\!]$ and $[\![\varepsilon]\!]$ to denote the starting symbol and empty word, respectively; and for each table $\tau \in T$ and each replacement rule $\tau\colon A \to B_1B_2 \cdots B_\ell$ with each $B_j \in V \cup \Sigma$, and for each $k \in \mathbb{N}$ with $1 \leqslant k \leqslant \ell$, we have $[\![B_kB_{k+1} \cdots B_\ell]\!]$ as a distinct symbol of Γ. For example, if $T = \{\alpha, \beta, \gamma\}$ where

$$\alpha\colon S \to ABC \mid BCB \qquad \beta\colon \begin{cases} A \to BA \mid B \\ B \to B \mid \varepsilon \\ C \to B \end{cases} \qquad \gamma\colon \begin{cases} A \to a \\ B \to bb\,, \\ C \to c \end{cases}$$

then the corresponding pushdown alphabet is given by

$$\Gamma = \{[\![S]\!], [\![\varepsilon]\!]\} \cup \underbrace{\{[\![ABC]\!], [\![BC]\!], [\![C]\!], [\![BCB]\!], [\![CB]\!], [\![B]\!]\}}_{\text{suffixes of rules in } \alpha} \cup \underbrace{\{[\![BA]\!], [\![A]\!], [\![B]\!]\}}_{\text{suffixes of rules in } \beta} \cup \underbrace{\{[\![a]\!], [\![bb]\!], [\![b]\!], [\![c]\!]\}}_{\text{suffixes of rules in } \gamma}.$$

Notice that the pushdown alphabet Γ is finite as an ET0L grammar can have only finitely many replacement rules.

The machine $\mathcal{M}$ has three states $\{q_0, q_{\text{apply}}, q_{\text{accept}}\} = Q$ where q_0 is the start state and q_{accept} is the only accepting state. The idea of state q_{apply} is that its transitions to itself emulate an application of a table, which it sees on the check-stack, to a non-terminal, which it sees on the pushdown.

Now that we have chosen our alphabets and states, we are ready to describe the transition relations, θ, of $\mathcal{M}$.

To begin a computation we have the transition

$$((q_0, \varepsilon, (\mathfrak{b}, \mathfrak{b})), (q_{\text{apply}}, [\![S]\!]\mathfrak{b}))$$

which pushes the start symbol of the grammar onto the pushdown (see Figure 2.6). In the remainder of this proof, we will ensure that $\mathcal{M}$ is only able to empty its pushdown by emulating a derivation of its input word with respect to the grammar G. Thus, we have the transition

$$((q_{\text{apply}}, \varepsilon, (\mathfrak{b}, \mathfrak{b})), (q_{\text{accept}}, \mathfrak{b})),$$

to accept when the pushdown is emptied.

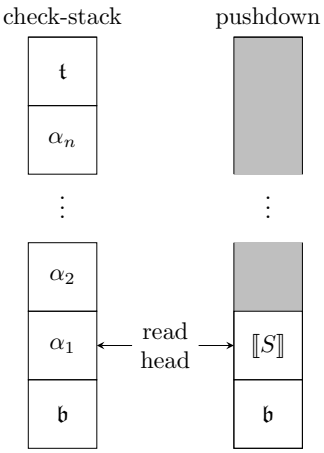

Figure 2.6: The stack configuration when the machine first enters state q_{apply}.

We will now describe how the transitions from q_{apply} performs a derivation in the grammar G.

Suppose that $\mathcal{M}$ is in state q_{apply}, then the remaining transitions can be separated into the following three cases.

1. *Applying a table.* Suppose that the read-head of $\mathcal{M}$ sees $(\tau, [\![A_1 A_2 \cdots A_m]\!])$ where τ is a table of the grammar and $m \geqslant 1$. Then, we want $\mathcal{M}$ to apply the table τ to the word $A_1 A_2 \cdots A_m$. We do this by applying τ from left-to-right, i.e., for each $B_1 B_2 \cdots B_k \in (V \cup \Sigma)^*$ such that $A_1 \to^\tau B_1 B_2 \cdots B_k$ we have transitions

   $$((q_{\text{apply}}, \varepsilon, (\tau, [\![A_1 A_2 \cdots A_m]\!])), (q_{\text{apply}}, [\![B_1 B_2 \cdots B_k]\!][\![A_2 \cdots A_m]\!]))$$

   to expand the letter A_1 to a nondeterministic choice of sequence $B_1 B_2 \cdots B_k$. See Figure 2.7 for an example of this expansion.

2. *Empty word.* Suppose that the read-head of $\mathcal{M}$ sees $(\tau, [\![\varepsilon]\!])$ where τ is a table. Then, since there is no way to expand ε any further, we pop this letter from the pushdown, i.e., for each table $\tau \in T$ we have a transition

$$((q_{\text{apply}}, \varepsilon, (\tau, [\![\varepsilon]\!])), (q_{\text{apply}}, \varepsilon)).$$

3. *Finished applying tables.* Suppose that $\mathcal{M}$ is at the top of its check-stack, i.e., its read-head sees $(\mathfrak{t}, [\![A_1 A_2 \cdots A_m]\!])$ where $m = 0$ corresponds to the read-head seeing $(\mathfrak{t}, [\![\varepsilon]\!])$. Then, we have no further tables to apply to $A_1 A_2 \cdots A_m$. Thus, each A_j must be a terminal letter of Σ, and we must see $A_1 A_2 \cdots A_m$ on the input tape. Thus, for each $[\![a_1 a_2 \cdots a_m]\!] \in \Gamma$ with each $a_j \in \Sigma$, we have a transition

$$((q_{\text{apply}}, a_1 a_2 \cdots a_m, (\mathfrak{t}, [\![a_1 a_2 \cdots a_m]\!])), (q_{\text{apply}}, \varepsilon)).$$

   Notice here that, if the letter on the pushdown does not correspond to a word in Σ^*, then we have no path to q_{accept} and thus we reject.

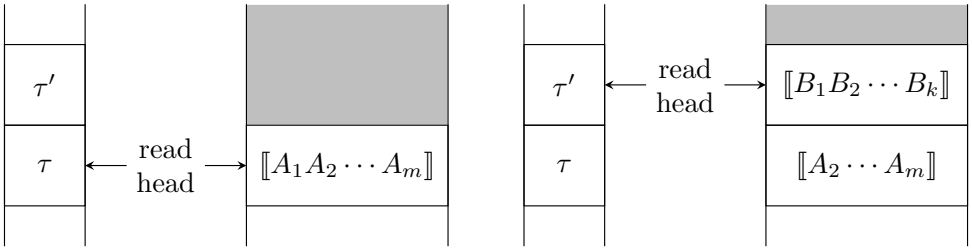

Figure 2.7: Expanding the letter A_1 with respect to the table τ.

Soundness and Completeness.

Suppose that $\mathcal{M}$ is given a word $w \in L$ on its input tape. Then, there must exist some $\alpha \in \mathcal{R}$ such that $S \to^{\alpha} w$ in the grammar G. Thus, $\mathcal{M}$ can nondeterministically choose a check-stack of $\alpha\mathfrak{t}$ and emulate a derivation of w from S as previously described. Hence, $\mathcal{M}$ will accept any word from L.

Suppose that $\mathcal{M}$ accepts a given word $w \in L$ with a check-stack of $\alpha\mathfrak{t}$. Then, by following the previous construction, it can be seen that we can recover a derivation $S \to^{\alpha} w$. Hence, $\mathcal{M}$ can only accept words in L.

Therefore, $\mathcal{M}$ accepts a given word if and only if it is in the language L. □

We now complete our proof of Proposition 2.21 by showing that the family of languages recognised by cspd automata forms a subclass of ET0L as follows.

Lemma 2.23. *A language recognised by a cspd automaton is ET0L.*

Proof. Let $\mathcal{M} = (Q, \Sigma, \Gamma, \Delta, \mathfrak{b}, \mathcal{R}, \theta, q_0, F)$ be a given cspd automaton, where we will assume without loss of generality that $\mathcal{M}$ satisfies Lemma 2.20.

We will construct a grammar $G = (\Sigma, V, T, \mathcal{R}', S)$ as in Lemma 2.19, which generates precisely the language recognised by $\mathcal{M}$.

Considering assumptions 2–4 from Lemma 2.20, a plot of the height of the pushdown during a successful computation of $\mathcal{M}$ (i.e. one that leads to the accepting state) will resemble a Dyck path; that is, the non-negative height of the pushdown is zero at the beginning and end of such a computation.

For each pair of states $p, q \in Q$ and each pushdown letter $g \in \Gamma$, the grammar G has a non-terminal letter $A^g_{p,q}$. The non-terminal $A^g_{p,q}$ corresponds to the situation where $\mathcal{M}$ has just pushed g onto its pushdown on a transition to the state p; and that when $\mathcal{M}$ pops this g, it will do so on a transition to the state q. Further, G has a non-terminal $A^{\mathfrak{b}}_{q_0,q_{\text{accept}}}$ which corresponds to any path from the initial configuration to the accepting state. (See Figure 2.8.) Thus, the starting symbol of G will be given by $S = A^{\mathfrak{b}}_{q_0,q_{\text{accept}}}$.

For each letter $c \in \Delta \cup \{\mathfrak{b}\}$ on the check-stack, we have a table $\tau_c \in T$ in the grammar G. Moreover, by taking a regular expression for the language $\mathfrak{b}\mathcal{R}$ and replacing each instance of $c \in \Delta \cup \{\mathfrak{b}\}$ with its corresponding table τ_c, we obtain the rational control $\mathcal{R}'$ of the grammar G.

Thus, in the remainder of this proof we describe the tables τ_c, and the way in which they emulate a computation of $\mathcal{M}$. Note that when describing these tables we make use of the notation introduced in Lemma 2.19; in particular, we will use replacements with regular languages on their right-hand sides.

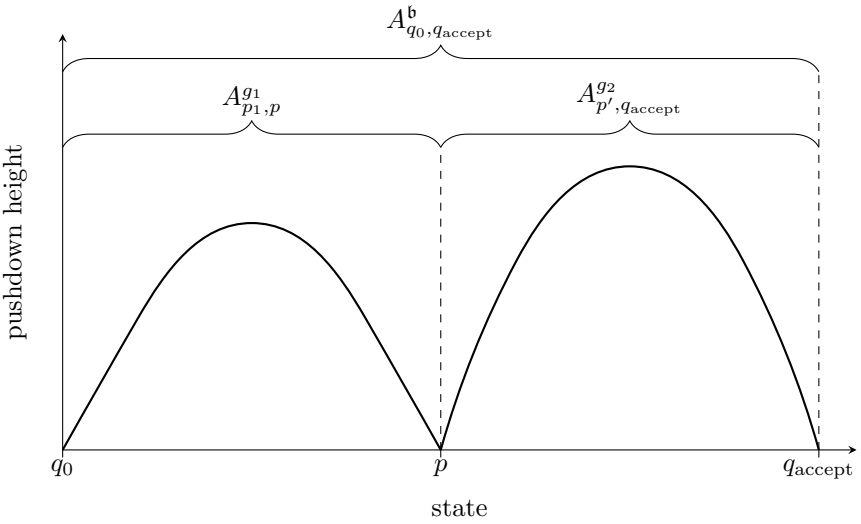

Figure 2.8: The height of the pushdown during an example computation.

For each $p, q \in Q$ and $(c, b) \in (\Delta \times \Gamma) \cup \{(\mathfrak{b}, \mathfrak{b})\}$, let $\mathcal{F}^{(c,b)}_{p,q}$ be a finite-state automaton. The idea here is that, with $\mathcal{L}^{(c,b)}_{p,q}$ as the regular language accepted by $\mathcal{F}^{(c,b)}_{p,q}$, we will have the replacement rule $\tau_c \colon A^b_{p,q} \to \mathcal{L}^{(c,b)}_{p,q}$.

The states of each $\mathcal{F}^{(c,b)}_{p,q}$ include all the states of $\mathcal{M}$, and an additional disjoint state λ. We denote these states as $Q' = Q \cup \{\lambda\}$ where Q are states of $\mathcal{M}$. The state λ is the only accepting state in each $\mathcal{F}^{(c,b)}_{p,q}$.

Let some $\mathcal{F}_{p,q}^{(c,b)}$ be given. Then, a given state $r \in Q \subseteq Q'$ of $\mathcal{F}_{p,q}^{(c,b)}$ corresponds to the situation where $\mathcal{M}$ is in state r and its read-head sees (c, b). Thus, the start state of $\mathcal{F}_{p,q}^{(c,b)}$ is given by $p \in Q'$. Further, the accepting state λ corresponds to the configuration of $\mathcal{M}$ immediately after following a path described by $A_{p,q}^{b}$.

With a finite-state automaton $\mathcal{F}_{p,q}^{(c,b)}$ given, we now describe its transitions. Suppose that $\mathcal{M}$ is in the state $r \in Q$ and its read-head sees (c, b), then $\mathcal{M}$ can either push some letter $x \in \Delta$ with a transition of the form

$$((r, \alpha_1\alpha_2 \cdots \alpha_k, (c, b)), (s, xb)) \tag{2.5}$$

then follow a path described by $A_{s,t}^{x}$ for some state $t \in Q$; or $\mathcal{M}$ can complete a path described by $A_{r,q}^{b}$ with a transition of the form

$$((r, \alpha_1\alpha_2 \cdots \alpha_k, (c, b)), (q, \varepsilon)) \quad \text{or} \quad ((r, \alpha_1\alpha_2 \cdots \alpha_k, (\mathfrak{b}, \mathfrak{b})), (q, \mathfrak{b})) \tag{2.6}$$

depending on whether q is the accepting state q_{accept} (see Lemma 2.20).

Thus, for each transition in $\mathcal{M}$ of form (2.5), and each state $t \in Q \subseteq Q'$, we have a transition in $\mathcal{F}_{p,q}^{(c,b)}$ from state r to t labelled $\alpha_1\alpha_2 \cdots \alpha_k A_{s,t}^{x}$.

Further, for each transition of form (2.6), we have a transition in $\mathcal{F}_{p,q}^{(c,b)}$ from state r to λ labelled $\alpha_1\alpha_2 \cdots \alpha_k$.

For each check-stack letter $c \in \Delta \cup \{\mathfrak{b}\}$ of $\mathcal{M}$ and non-terminal $A_{p,q}^{b}$ of G, we define the tables τ_c such that $\tau_c \colon A_{p,q}^{b} \to \mathcal{L}_{p,q}^{(c,b)}$ where $\mathcal{L}_{p,q}^{(c,b)}$ is the regular language recognised by $\mathcal{F}_{p,q}^{(c,b)}$. Since regular language is a subset of ET0L, then, by Lemma 2.19, the grammar G produces an ET0L language as required.

Soundness and Completeness.

Suppose that $\mathcal{M}$ is able to accept the word $w \in \Sigma^*$ with $\alpha \in \mathcal{R}$ chosen as its check-stack. Then, by following such a computation to the accepting state, we can construct a derivation $S \to^{\mathfrak{b}\alpha} w$ in the grammar G. Thus, every word that is accepted by $\mathcal{M}$ is in the language produce by G.

Let $w \in L(G)$ be a word produced by the grammar G. Then, there must exist some sequence of tables $\beta \in \mathcal{R}'$ such that $S \to^{\beta} w$; and thus, for any corresponding derivation in the grammar G, and by following our construction, we can recover a computation of $\mathcal{M}$ which accepts w.

Therefore, G generates precisely the language that is recognised by $\mathcal{M}$. □

2.6. Concluding Remarks

In this chapter we defined and motivated several families of formal languages and classes of power series. In Chapter 3 we make use of the theory of ET0L grammars, and their equivalence to cspd automata to prove Theorem C. In Chapter 4, we return to formal language theory and define the family of *polyhedrally constrained languages*. We then use this family of languages in the proof of Theorem A in which we show that every virtually abelian group has a holonomic generating function.

Chapter 3

Co-Word Problems

Let G be a group with a finite monoid generating set S. Then, if we are given two words $u, v \in S^*$, it is a natural question to ask whether these words represent the same group element. This question is equivalent to asking if the word $w = uv^{-1} \in S^*$ represents the group identity. We then define the formal language

$$\mathrm{WP}_S = \{w \in S^* \mid \overline{w} = 1\}$$

which we refer to as the *word problem* with respect to the generating set S.

For each finite monoid generating set S of G, we see that $\langle S \mid \mathrm{WP}_S \rangle$ is a presentation for G. Thus, the word problem completely describes a group. A classification of the word problem is thus one method to characterise the complexity of a group.

The study of the formal language complexity of group word problems began with Anisimov [2] who showed that the word problem is regular if and only if the group is finite. This was extended by Muller and Schupp [78] who showed that a group has a context-free word problem if and only if it is virtually free, in which case, the word problem is deterministic context-free. It was shown by Elder, Kambites and Ostheimer [42] that a group has a word problem that is (deterministic) blind multicounter if and only if it is virtually abelian. It was shown by Holt, Rees and Shapiro [62] that a group has a *growing context-sensitive* word problem if and only if its word problem can be solved by a certain generalisation of *Dehn's algorithm*, however, this result does not appear to provide a group-theoretic classification.

An alternate method of obtaining characterisation of group word problems is to characterise the *co-word problem* $\mathrm{coWP}_S = S^* \setminus \mathrm{WP}_S$, that is, the formal language

$$\mathrm{coWP}_S = \{w \in S^* \mid \overline{w} \neq 1\}\,.$$

It is known that regular and deterministic context-free languages are closed under taking set complements, see [85, Theorem 8.4] and [92, Theorem 2.42], respectively. Thus, the co-word problem for a group is

- regular if and only if the group is finite; and

- deterministic context-free if and only if the group is virtually free.

The class of groups for which coWP_S is context-free was first studied by Holt, Rees, Röver and Thomas [61], in particular, they showed that a polycyclic group has a context-free co-word problem if and only if it is virtually abelian. The study of co-context-free groups was later extended by Lehnert and Schweitzer [70] who showed that Thompson's group V has a context-free co-word problem. It is conjectured [17, Conjecture 5] that a group has a context-free co-word problem if and only if it is a subgroup of Thompson's group V. However, a potential counter-example to such a classification is provided in [11].

It was shown by Holt and Röver [63] that the class of *bounded automata groups* have co-word problems that can be recognised as *indexed languages*, as defined by Aho [1]. The class of bounded automata groups includes important examples such as Grigorchuk's group of intermediate growth, the Gupta-Sidki groups, and many more [55, 57, 79, 91].

ET0L languages form a proper subfamily of the indexed languages (see Corollary 4.1 in [36] and Proposition 4.5 in [40]). For the specific case of the Grigorchuk group, Ciobanu, Elder and Ferov [32] constructed an ET0L grammar for the co-word problem. In this chapter, we generalise this result by showing that the co-word problem for any bounded automata group is ET0L. In particular, for each bounded automata group, we construct a cspd automaton which recognises the language of geodesics.

3.1. Generating Sets

Many interesting families of formal language are closed under inverse word homomorphism, as defined in Definition 3.1. For example, the class of regular, context-free and ET0L languages have this closure property.

Definition 3.1. *A family of formal languages $\mathcal{F}$ is* closed under inverse word homomorphism*, if for each language $L \in \mathcal{F} \subseteq \Sigma^*$, and each monoid homomorphism $h\colon \Gamma^* \to \Sigma^*$ where Γ is an alphabet, the language $h^{-1}(L) = \{w \in \Gamma^* \mid h(w) \in L\}$ lies within $\mathcal{F}$.*

We then see that the formal language complexity of the co-word problem for a group is well defined in the following sense.

Lemma 3.2. *Let G be a group with a finite monoid generating set S. If $\mathrm{coWP}_S \in \mathcal{F}$ and $\mathcal{F}$ is closed under inverse word homomorphism, then $\mathrm{coWP}_X \in \mathcal{F}$ for each finite monoid generating set X of G.*

Proof. For each $x \in X$ we choose a word $w_x \in S^*$ such that $\overline{x} = \overline{w_x}$. We then define a monoid homomorphism $h\colon X^* \to S^*$ where $h(x) = w_x$ for each $x \in X$. We see that $\mathrm{coWP}_X = h^{-1}(\mathrm{coWP}_S)$, and thus $\mathrm{coWP}_X \in \mathcal{F}$ as required. □

Let $\mathcal{F}$ be a family of languages which is closed under inverse word homomorphism. Then, a group is co-$\mathcal{F}$ if its co-word problem lies in the class $\mathcal{F}$, for any and thus all generating sets. It was shown by Čulik [36, Corollary 3.2 on p. 40] that ET0L languages form a *full AFL*, one of the defining properties of this being closure with respect to inverse word homomorphism. From Lemma 3.2 it is well defined to ask if a group is co-ET0L.

3.2. Bounded Automata Groups

In this section, we define the class of *bounded automata groups*. Each such group is a group of automorphisms of an infinite rooted tree and can be completely described using finitely many finite-state rewrite automata. We begin by defining rooted trees as follows.

For $d \geqslant 2$, let $\mathcal{T}_d$ denote the d-regular rooted tree, that is, the infinite rooted tree where each vertex has exactly d children. We identify the vertices of $\mathcal{T}_d$ with words in Σ^* where $\Sigma = \{a_1, a_2, \ldots, a_d\}$. In particular, we identify the root with the empty word $\varepsilon \in \Sigma^*$ and we identify the k-th child of each vertex $v \in \mathrm{V}(\mathcal{T}_d)$ with the word va_k, see Figure 3.1.

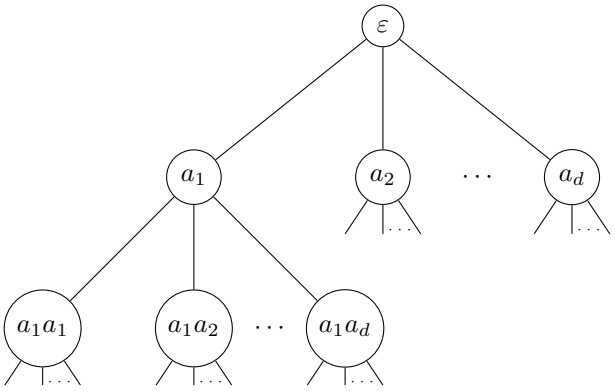

Figure 3.1: A labelling of the vertices of $\mathcal{T}_d$.

Recall that an automorphism of a graph is a bijective mapping of the vertex set that preserves adjacency, thus an automorphism of $\mathcal{T}_d$ preserves the root and levels of the tree. We denote the group of automorphisms of $\mathcal{T}_d$ as $\mathrm{Aut}(\mathcal{T}_d)$. We write $\mathrm{Sym}(\Sigma)$ for the *permutation group of* Σ. An important observation is that $\mathrm{Aut}(\mathcal{T}_d)$ can be seen as the wreath product $\mathrm{Aut}(\mathcal{T}_d) \wr \mathrm{Sym}(\Sigma)$, since any automorphism $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ can be written uniquely as $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d) \cdot \sigma$ where each $\alpha_i \in \mathrm{Aut}(\mathcal{T}_d)$ is an automorphism of the sub-tree with root a_i, and $\sigma \in \mathrm{Sym}(\Sigma)$ is a permutation of the first level.

Let $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d) \cdot \sigma \in \mathrm{Aut}(\mathcal{T}_d) \wr \mathrm{Sym}(\Sigma)$. Then, for any letter $a_i \in \Sigma$, the *restriction of* α *to* a_i, denoted $\alpha|_{a_i} = \alpha_i$, is the action of α on the sub-tree with root a_i (which is given by α_i). Given any vertex $w = w_1 w_2 \cdots w_k \in \Sigma^*$ of $\mathcal{T}_d$, we can define the *restriction of* α *to* w recursively as

$$\alpha|_w = \left(\alpha|_{w_1 w_2 \cdots w_{k-1}}\right)\Big|_{w_k}$$

and thus describe the action of α on the sub-tree with root w.

The action of each element of a bounded automata group on its associated tree can be described using a certain type of finite-state rewrite automata, which we will refer to as a Σ-automaton. We define this class of automata as follows.

Definition 3.3. *A* Σ-automaton, (Γ, v), *is a finite directed graph with a distinguished vertex* v*, called the initial state, and a* $(\Sigma \times \Sigma)$*-labelling of its edges, such that each vertex*

has exactly $|\Sigma|$ *outgoing edges; and for each* $a \in \Sigma$ *each vertex has exactly one incoming edge of the form* (a, a') *and exactly one outgoing edge of the form* (a', a)*. Thus, the outgoing edges define a permutation of* Σ*.*

From a Σ-automaton, we may then define a tree automorphism as follows.

Definition 3.4. *Let* (Γ, v) *be a* Σ*-automaton with* $\Sigma = \{a_1, \ldots, a_d\}$*, then we define an automorphism* $\alpha_{(\Gamma,v)} \in \mathrm{Aut}(\mathcal{T}_d)$ *as follows. Notice that for each vertex* $b_1 b_2 \cdots b_k \in \mathrm{V}(\mathcal{T}_d) = \Sigma^*$*, there is a unique path in the graph* Γ *starting from the initial vertex,* v*, of the form* $(b_1, b_1')\,(b_2, b_2') \cdots (b_k, b_k')$*. We define* $\alpha_{(\Gamma,v)}$ *such that* $\alpha_{(\Gamma,v)}(b_1 b_2 \cdots b_k) = b_1' b_2' \cdots b_k'$*. From the definition of* Σ*-automata it the follows that* $\alpha_{(\Gamma,v)}$ *is an isomorphism.*

We provide an example of a Σ-automaton in Figure 3.2.

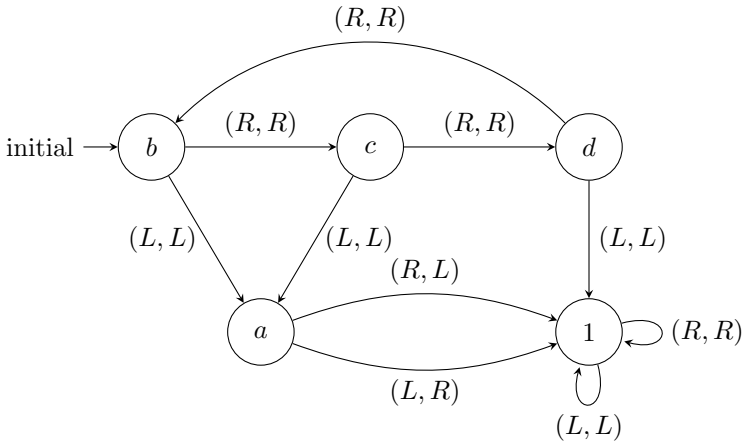

Figure 3.2: A Σ-automaton for the generator b in Grigorchuk's group.

An *automaton automorphism*, α, of the tree $\mathcal{T}_d$ is an automorphism for which there exists a Σ-automaton, (Γ, v), such that $\alpha = \alpha_{(\Gamma,v)}$. We write $\mathcal{A}(\mathcal{T}_d)$ for the set of all automata automorphisms of the tree $\mathcal{T}_d$. The set $\mathcal{A}(\mathcal{T}_d)$ forms a group [91, Proposition 1]. Moreover, a subgroup of $\mathcal{A}(\mathcal{T}_d)$ is called an *automata group*.

An automorphism $\alpha \in \mathrm{Aut}(\mathcal{T}_d)$ will be called *bounded* (originally defined in [91]) if there exists a constant $N \in \mathbb{N}$ such that for each $k \in \mathbb{N}$, there are no more than N vertices $v \in \Sigma^*$ with $|v| = k$ (i.e. at level k) such that $\alpha|_v \neq 1$. Thus, the action of such a bounded automorphism will, on each level, be trivial on all but (up to) N sub-trees. The set of all such automorphisms form a group which we will denote as $\mathcal{B}(\mathcal{T}_d)$. The group of all *bounded automaton automorphisms* is defined as the intersection $\mathcal{A}(\mathcal{T}_d) \cap \mathcal{B}(\mathcal{T}_d)$, which we will denote as $\mathcal{D}(\mathcal{T}_d)$. A subgroup of $\mathcal{D}(\mathcal{T}_d)$ is called a *bounded automata group*.

A *finitary automorphism* of $\mathcal{T}_d$ is an automorphism ϕ such that there exists a constant $k \in \mathbb{N}$ for which $\phi|_v = 1$ for each $v \in \Sigma^*$ with $|v| = k$. Thus, a finitary automorphism is one that is trivial after some k levels of the tree. Given a finitary automorphism ϕ, the smallest k for which this definition holds will be called its *depth* and will be denoted as $\mathrm{depth}(\phi)$. We will denote the group formed by all finitary automorphisms of $\mathcal{T}_d$ as

$\mathrm{Fin}(\mathcal{T}_d)$. See Figure 3.3 for examples of the actions of finitary automorphisms on their associated trees (where any unspecified sub-tree is fixed by the action).

Figure 3.3: Examples of finitary automorphisms $a, b \in \mathrm{Fin}(\mathcal{T}_2)$.

Let $\delta \in \mathcal{A}(\mathcal{T}_d) \setminus \mathrm{Fin}(\mathcal{T}_d)$. We call δ a *directed automaton automorphism* if

$$\delta = (\phi_1, \phi_2, \ldots, \phi_{k-1}, \delta', \phi_{k+1}, \ldots, \phi_d) \cdot \sigma \in \mathrm{Aut}(\mathcal{T}_d) \wr \mathrm{Sym}(\Sigma) \tag{3.1}$$

where each ϕ_j is finitary and δ' is also directed automaton (that is, not finitary and can also be written in this form). We call $\mathrm{dir}(\delta) = b = a_k \in \Sigma$, where $\delta' = \delta|_b$ is directed automaton, the *direction* of δ; and we define the *spine* of δ, denoted $\mathrm{spine}(\delta) \in \Sigma^\omega$, recursively such that $\mathrm{spine}(\delta) = \mathrm{dir}(\delta)\,\mathrm{spine}(\delta')$. We denote the set of all directed automaton automorphisms as $\mathrm{Dir}(\mathcal{T}_d)$. See Figure 3.4 for examples of directed automaton automorphisms (in which a and b are the finitary automorphisms in Figure 3.3).

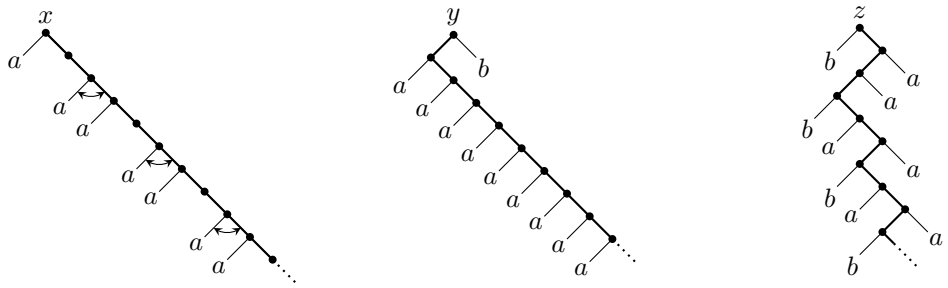

Figure 3.4: Examples of directed automorphisms $x, y, z \in \mathrm{Dir}(\mathcal{T}_2)$.

The following lemma is essential to prove our main theorem.

Lemma 3.5 (Lemma 3 in [63])**.** *The spine,* $\mathrm{spine}(\delta) \in \Sigma^\omega$*, of a directed automaton automorphism,* $\delta \in \mathrm{Dir}(\mathcal{T}_d)$*, is eventually periodic, that is, there exists some* $\iota = \iota_1\iota_2\cdots\iota_s \in \Sigma^*$*, called the* initial section*, and* $\pi = \pi_1\pi_2\cdots\pi_t \in \Sigma^*$*, called the* periodic section*, such that* $\mathrm{spine}(\delta) = \iota\,\pi^\omega$*; and*

$$\delta|_{\iota\,\pi^k\,\pi_1\pi_2\cdots\pi_j} = \delta|_{\iota\,\pi_1\pi_2\cdots\pi_j} \tag{3.2}$$

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

Proof. Let (Γ, v) be a Σ-automaton such that $\delta = \alpha_{(\Gamma,v)}$. By the definition of Σ-automata, for any given vertex $w = w_1w_2\cdots w_k \in \Sigma^*$ of $\mathcal{T}_d$ there exists a vertex $v_w \in \mathrm{V}(\Gamma)$ such that $\delta|_w = \alpha_{(\Gamma,v_w)}$. In particular, such a vertex v_w can be obtained by following the path

with edges labelled $(w_1, w_1')(w_2, w_2') \cdots (w_k, w_k')$. Then, since there are only finitely many vertices in Γ, the set of all restrictions of δ is finite, that is,

$$\#\left\{ \delta|_w = \alpha_{(\Gamma, v_w)} \mid w \in \Sigma^* \right\} < \infty. \tag{3.3}$$

Let $b = b_1 b_2 b_3 \cdots = \operatorname{spine}(\delta) \in \Sigma^\omega$ denote the spine of δ. Then, there exists some $n, m \in \mathbb{N}$ with $n < m$ such that

$$\delta|_{b_1 b_2 \cdots b_n} = \delta|_{b_1 b_2 \cdots b_n \cdots b_m} \tag{3.4}$$

as otherwise there would be infinitely many distinct restrictions of the form $\delta|_{b_1 b_2 \cdots b_k}$ thus contradicting (3.3). By the definition of the spine, it follows that

$$\operatorname{spine}\left(\delta|_{b_1 b_2 \cdots b_n}\right) = (b_{n+1} b_{n+2} \cdots b_m) \operatorname{spine}\left(\delta|_{b_1 b_2 \cdots b_n \cdots b_m}\right).$$

Hence, by (3.4),

$$\operatorname{spine}\left(\delta|_{b_1 b_2 \cdots b_n}\right) = (b_{n+1} b_{n+2} \cdots b_m)^\omega.$$

Thus,

$$\begin{aligned} \operatorname{spine}(\delta) &= (b_1 b_2 \cdots b_n) \operatorname{spine}\left(\delta|_{b_1 b_2 \cdots b_n}\right) \\ &= (b_1 b_2 \cdots b_n) \, (b_{n+1} b_{n+2} \cdots b_m)^\omega. \end{aligned}$$

By taking $\iota = b_1 b_2 \cdots b_n$ and $\pi = b_{n+1} b_{n+2} \cdots b_m$, we have $\operatorname{spine}(\delta) = \iota \, \pi^\omega$. Moreover, from (3.4), we have equation (3.2) as required. □

Notice that each finitary and directed automata automorphism is also bounded, in fact, we have the following proposition which shows that the generators of any given bounded automata group can be written as words in $\operatorname{Fin}(\mathcal{T}_d)$ and $\operatorname{Dir}(\mathcal{T}_d)$.

Proposition 3.6 (Proposition 16 in [91])**.** *The group $\mathcal{D}(\mathcal{T}_d)$ of bounded automata automorphisms is generated by $\operatorname{Fin}(\mathcal{T}_d)$ together with $\operatorname{Dir}(\mathcal{T}_d)$.*

3.2.1. Co-Word Problems

We may now prove the following characterisation of bounded automata groups.

Theorem D. *Every finitely-generated bounded automata group is co-ET0L.*

The idea of the proof is straightforward: we construct a cspd machine that nondeterministically chooses a vertex $v \in \mathrm{V}(\mathcal{T}_d)$, writing its labels on the check-stack and a copy on its pushdown; as it reads letters from input, it uses the pushdown to keep track of where the chosen vertex is moved; and finally it checks whether the pushdown and the check-stack differ. The full details are as follows.

Proof. Let $G \subseteq \mathcal{D}(\mathcal{T}_d)$ be a bounded automata group with finite monoid generating set X. By Proposition 3.6, we can define a map

$$\varphi \colon X \to (\operatorname{Fin}(\mathcal{T}_d) \cup \operatorname{Dir}(\mathcal{T}_d))^*$$

so that x and $\varphi(x)$ are equal in $\mathcal{D}(\mathcal{T}_d)$ for each $x \in X$. Let

$$Y = \left\{\alpha \in \mathrm{Fin}(\mathcal{T}_d) \cup \mathrm{Dir}(\mathcal{T}_d) \;\middle|\; \alpha \text{ or } \alpha^{-1} \text{ is a letter in } \varphi(x) \text{ for some } x \in X\right\}$$

which is a finite generating set for a group which contains G as a subgroup. Consider the group $H \subseteq \mathcal{D}(\mathcal{T}_d)$ generated by Y. Since ET0L is closed under inverse word homomorphism, it suffices to prove that coWP_Y is ET0L, as coWP_X is its inverse image under the mapping $X^* \to Y^*$ induced by φ. We construct a cspd machine $\mathcal{M}$ that recognises coWP_Y, thus proving that G is co-ET0L.

Let $\alpha = \alpha_1\alpha_2\cdots\alpha_n \in Y^*$ denote an input word given to $\mathcal{M}$. The execution of the cspd will be separated into four stages; (1) choosing a vertex $v \in \Sigma^*$ of $\mathcal{T}_d$ which witnesses the non-triviality of α (and placing it on the stacks); (2a) reading a finitary automorphism from the input tape; (2b) reading a directed automaton automorphism from the input tape; and (3) checking that the action of α on v that it has computed is non-trivial.

After Stage 1, $\mathcal{M}$ will be in state q_{comp}. From here, $\mathcal{M}$ nondeterministically decides to either read from its input tape, performing either Stage 2a or 2b and returning to state q_{comp}; or to finish reading from input by performing Stage 3.

We set both the check-stack and pushdown alphabets to be $\Sigma \cup \{\mathfrak{t}\}$, i.e., we have $\Delta = \Gamma = \Sigma \cup \{\mathfrak{t}\}$. The letter $\mathfrak{t}$ will represent the top of the check-stack.

Stage 1: choosing a witness $v = v_1v_2\cdots v_m \in \Sigma^*$.

If α is non-trivial, then there must exist a vertex $v \in \Sigma^*$ such that $\alpha \cdot v \neq v$. Thus, we nondeterministically choose such a witness from $\mathcal{R} = \Sigma^*\mathfrak{t}$ and store it on the check-stack, where the letter $\mathfrak{t}$ represents the top of the check-stack.

From the start state, q_0, $\mathcal{M}$ will copy the contents of the check-stack onto the pushdown, then enter the state $q_{\mathrm{comp}} \in Q$. Formally, this will be achieved by adding the transitions (for each $a \in \Sigma$):

$$((q_0, \varepsilon, (\mathfrak{b}, \mathfrak{b})), (q_0, \mathfrak{t}\mathfrak{b})),\ ((q_0, \varepsilon, (a, \mathfrak{t})), (q_0, \mathfrak{t}a)),\ ((q_0, \varepsilon, (\mathfrak{t}, \mathfrak{t})), (q_{\mathrm{comp}}, \mathfrak{t})).$$

This stage concludes with $\mathcal{M}$ in state q_{comp}, and the read-head pointing to $(\mathfrak{t}, \mathfrak{t})$. Note that whenever the machine is in state q_{comp} and $\alpha_1\alpha_2\cdots\alpha_k$ has been read from input, then the contents of pushdown will represent the permuted vertex $(\alpha_1\alpha_2\cdots\alpha_k)\cdot v$. Thus, the two stacks are initially the same as no input has been read and thus no group action has been simulated. In Stages 2a and 2b, only the height of the check-stack is important, that is, the exact contents of the check-stack will become relevant in Stage 3.

Stage 2a: reading a finitary automorphism $\phi \in Y \cap \mathrm{Fin}(\mathcal{T}_d)$.

By definition, there exists some $k_\phi = \mathrm{depth}(\phi) \in \mathbb{N}$ such that $\phi|_u = 1$ for each $u \in \Sigma^*$ for which $|u| \geqslant k_\phi$. Thus, given a vertex $v = v_1v_2\cdots v_m \in \Sigma^*$, we have

$$\phi(v) = \phi(v_1v_2\cdots v_{k_\phi})\ v_{(k_\phi+1)}\cdots v_m.$$

Given that $\mathcal{M}$ is in state q_{comp} with $\mathfrak{t}v_1v_2\cdots v_m\mathfrak{b}$ on its pushdown, we will read ϕ from input, move to state $q_{\phi,\varepsilon}$ and pop the $\mathfrak{t}$; we will then pop the next k_ϕ (or fewer if $m < k_\phi$) letters off the pushdown, and as we are popping these letters we visit the sequence of

states q_{ϕ,v_1}, q_{ϕ,v_1v_2}, $\ldots$, $q_{\phi,v_1v_2\cdots v_{k_\phi}}$. From the final state in this sequence, we then push $\mathfrak{t}\phi(v_1\cdots v_{k_\phi})$ onto the pushdown, and return to the state q_{comp}.

Formally, for letters $a, b \in \Sigma$, $\phi \in Y \cap \text{Fin}(\mathcal{T}_d)$, and vertices $u, w \in \Sigma^*$ where $|u| < k_\phi$ and $|w| = k_\phi$, we have the transitions

$$((q_{\text{comp}}, \phi, (\mathfrak{t}, \mathfrak{t})), (q_{\phi,\varepsilon}, \varepsilon)),\ ((q_{\phi,u}, \varepsilon, (a, b)), (q_{\phi,ub}, \varepsilon)),$$

$$((q_{\phi,w}, \varepsilon, (\varepsilon, \varepsilon)), (q_{\text{comp}}, \mathfrak{t}\phi(w)))$$

for the case where $m > k_\phi$, and

$$((q_{\phi,u}, \varepsilon, (\mathfrak{b}, \mathfrak{b})), (q_{\text{comp}}, \mathfrak{t}\phi(u)\mathfrak{b}))$$

for the case where $m \leqslant k_\phi$. Notice that we have finitely many states and transitions since Y, Σ and each k_ϕ is finite.

Stage 2b: reading a directed automorphism $\delta \in Y \cap \text{Dir}(\mathcal{T}_d)$.

By Lemma 3.5, there exists some $\iota = \iota_1\iota_2\cdots\iota_s \in \Sigma^*$ and $\pi = \pi_1\pi_2\cdots\pi_t \in \Sigma^*$ such that $\text{spine}(\delta) = \iota\,\pi^\omega$ and

$$\delta(\iota\pi^\omega) = I_1I_2\cdots I_s\ (\Pi_1\Pi_2\cdots\Pi_t)^\omega$$

where

$$I_i = \delta|_{\iota_1\iota_2\cdots\iota_{i-1}}(\iota_i) \qquad \text{and} \qquad \Pi_j = \delta|_{\iota\,\pi_1\pi_2\cdots\pi_{j-1}}(\pi_j).$$

Given some vertex $v = v_1v_2\cdots v_m \in \Sigma^*$, let $\ell \in \mathbb{N}$ be largest such that $p = v_1v_2\cdots v_\ell$ is a prefix of the sequence $\iota\pi^\omega = \text{spine}(\delta)$. Then by definition of directed automorphism, $\delta' = \delta|_p$ is directed and $\phi = \delta|_a$, where $a = v_\ell$, is finitary. Then, either $p = \iota_1\iota_2\cdots\iota_\ell$ and

$$\delta(v) = (I_1I_2\cdots I_\ell)\ \delta'(a)\ \phi(v_{\ell+2}v_{\ell+3}\cdots v_m),$$

or $p = \iota\pi^k\pi_1\pi_2\cdots\pi_j$, with $\ell = |\iota| + k\cdot|\pi| + j$, and

$$\delta(v) = (I_1I_2\cdots I_s)\,(\Pi_1\Pi_2\cdots\Pi_t)^k\,(\Pi_1\Pi_2\cdots\Pi_j)\ \delta'(a)\ \phi(v_{\ell+2}v_{\ell+3}\cdots v_m).$$

Hence, from state q_{comp} with $\mathfrak{t}v_1v_2\cdots v_m\mathfrak{b}$ on its pushdown, $\mathcal{M}$ reads δ from input, moves to state $q_{\delta,\iota,0}$ and pops the $\mathfrak{t}$; it then pops pa off the pushdown, using states to remember the letter a and the part of the prefix to which the final letter of p belongs (i.e. ι_i or π_j). From here, $\mathcal{M}$ performs the finitary automorphism ϕ on the remainder of the pushdown (using the same construction as Stage 2a), then, in a sequence of transitions, pushes $\mathfrak{t}\delta(p)\delta'(a)$ and returns to state q_{comp}. The key idea here is that, using only the knowledge of the letter a, the part of ι or π to which the final letter of p belongs, and the height of the check-stack, that $\mathcal{M}$ is able to recover $\delta(p)\delta'(a)$.

We now give the details of the states and transitions involved in this stage of the construction.

We have states $q_{\delta,\iota,i}$ and $q_{\delta,\pi,j}$ with $0 \leqslant i \leqslant |\iota|$, $1 \leqslant j \leqslant |\pi|$; where $q_{\delta,\iota,i}$ represents that the word $\iota_1\iota_2\cdots\iota_i$ has been popped off the pushdown, and $q_{\delta,\pi,j}$ represents that a word

$\iota\pi^k\pi_1\pi_2\cdots\pi_j$ for some $k\in\mathbb{N}$ has been popped of the pushdown. Thus, we begin with the transition

$$((q_{\text{comp}},\delta,(\mathfrak{t},\mathfrak{t})),(q_{\delta,\iota,0},\varepsilon)),$$

then for each $i,j\in\mathbb{N}$, $a\in\Sigma$ with $0\leqslant i<|\iota|$ and $1\leqslant j<|\pi|$, we have transitions

$$((q_{\delta,\iota,i},\varepsilon,(a,\iota_{i+1})),(q_{\delta,\iota,(i+1)},\varepsilon)),\quad((q_{\delta,\iota,|\iota|},\varepsilon,(a,\pi_1)),(q_{\delta,\pi,1},\varepsilon)),$$
$$((q_{\delta,\pi,j},\varepsilon,(a,\pi_{j+1})),(q_{\delta,\pi,(j+1)},\varepsilon)),\quad((q_{\delta,\pi,|\pi|},\varepsilon,(a,\pi_1)),(q_{\delta,\pi,1},\varepsilon))$$

to consume the prefix p.

After this, $\mathcal{M}$ will either be at the bottom of its stacks, or its read-head will see a letter on the pushdown that is not the next letter in the spine of δ. Thus, for each $i,j\in\mathbb{N}$ with $0\leqslant i\leqslant|\iota|$ and $1\leqslant j\leqslant|\pi|$ we have states $q_{\delta,\iota,i,a}$ and $q_{\delta,\pi,j,a}$; and for each $b\in\Sigma$ we have transitions

$$((q_{\delta,\iota,i},\varepsilon,(b,a)),(q_{\delta,\iota,i,a},\varepsilon))$$

where $a\neq\iota_{i+1}$ when $i<|\iota|$ and $a\neq\pi_1$ otherwise, and

$$((q_{\delta,\pi,j},\varepsilon,(b,a)),(q_{\delta,\pi,j,a},\varepsilon))$$

where $a\neq\pi_{j+1}$ when $j<|\pi|$ and $a\neq\pi_1$ otherwise.

Hence, after these transitions, $\mathcal{M}$ has consumed pa from its pushdown and will either be at the bottom of its stacks in some state $q_{\delta,\iota,i}$ or $q_{\delta,\pi,j}$; or will be in some state $q_{\delta,\iota,i,a}$ or $q_{\delta,\pi,j,a}$. Note here that, if $\mathcal{M}$ is in the state $q_{\delta,\iota,i,a}$ or $q_{\delta,\pi,j,a}$, then from Lemma 3.5 we know $\delta'=\delta|_p$ is equivalent to $\delta|_{\iota_1\iota_2\cdots\iota_i}$ or $\delta|_{\iota\pi_1\pi_2\cdots\pi_j}$, respectively; and further, we know the finitary automorphism $\phi=\delta|_{pa}=\delta'|_a$.

Thus, for each state $q_{\delta,\iota,i,a}$ and $q_{\delta,\pi,a}$ we will follow a similar construction to Stage 2a, to perform the finitary automorphism ϕ to the remaining letters on the pushdown, then push $\delta'(a)$ and return to the state $r_{\delta,\iota,i}$ or $r_{\delta,\pi,j}$, respectively. For the case where $\mathcal{M}$ is at the bottom of its stacks we have transitions

$$((q_{\delta,\iota,i},\varepsilon,(\mathfrak{b},\mathfrak{b})),(r_{\delta,\iota,i},\mathfrak{b})),\quad((q_{\delta,\pi,i},\varepsilon,(\mathfrak{b},\mathfrak{b})),(r_{\delta,\pi,i},\mathfrak{b}))$$

with $0\leqslant i\leqslant|\iota|$, $1\leqslant j\leqslant|\pi|$.

Thus, after following these transitions, $\mathcal{M}$ is in some state $r_{\delta,\iota,i}$ or $r_{\delta,\pi,j}$ and all that remains is for $\mathcal{M}$ to push $\delta(p)$ with $p=\iota_1\iota_2\cdots\iota_i$ or $p=\iota\pi^k\pi_1\pi_2\cdots\pi_k$, respectively, onto its pushdown. Thus, for each $i,j\in\mathbb{N}$ with $0\leqslant i\leqslant|\iota|$ and $1\leqslant j\leqslant|\pi|$, we have transitions

$$((r_{\delta,\pi,i},\varepsilon,(\varepsilon,\varepsilon)),(q_{\text{comp}},\mathfrak{t}I_1I_2\cdots I_i)),\quad((r_{\delta,\pi,j},\varepsilon,(\varepsilon,\varepsilon)),(r_{\delta,\pi},\Pi_1\Pi_2\cdots\Pi_j))$$

where from the state $r_{\delta,\pi}$, through a sequence of transitions, $\mathcal{M}$ will push the remaining $I\Pi^k$ onto the pushdown. In particular, we have transitions

$$((r_{\delta,\pi},\varepsilon,(\varepsilon,\varepsilon)),(r_{\delta,\pi},\Pi)),\quad((r_{\delta,\pi},\varepsilon,(\varepsilon,\varepsilon)),(q_{\text{comp}},\mathfrak{t}I)),$$

so that $\mathcal{M}$ can nondeterministically push some number of Π's followed by $\mathfrak{t}I$ before it finishes this stage of the computation. We can assume that the machine pushes the correct

number of Π's onto its pushdown as otherwise it will not see $\mathfrak{t}$ on its check-stack while in state q_{comp} and thus would not be able to continue with its computation, as every subsequent stage (2a,2b,3) of the computation begins with the read-head pointing to $\mathfrak{t}$ on both stacks.

Once again it is clear that this stage of the construction requires only finitely many states and transitions.

Stage 3: checking that the action is non-trivial.

At the beginning of this stage, the contents of the check-stack represent the chosen witness, v, and the contents of the pushdown represent the action of the input word, α, on the witness, i.e., $\alpha \cdot v$.

In this stage $\mathcal{M}$ checks if the contents of its check-stack and pushdown differ. Formally, we have states q_{accept} and q_{check}, with q_{accept} accepting; for each $a \in \Sigma$, we have transitions

$$((q_{\text{comp}}, \varepsilon, (\mathfrak{t}, \mathfrak{t})), (q_{\text{check}}, \varepsilon)), \quad ((q_{\text{check}}, \varepsilon, (a, a)), (q_{\text{check}}, \varepsilon))$$

to pop identical entries of the pushdown; and for each $(a, b) \in \Sigma \times \Sigma$ with $a \neq b$ we have a transition

$$((q_{\text{check}}, \varepsilon, (a, b)), (q_{\text{accept}}, \varepsilon))$$

to accept if the stacks differ by a letter.

Observe that if the two stacks are identical, then there is no path to the accepting state, q_{accept}, and thus $\mathcal{M}$ will reject. Notice also that by definition of cspd automata, if $\mathcal{M}$ moves into q_{check} before all input has been read, then $\mathcal{M}$ will not accept, i.e., an accepting state is only effective if all input is consumed.

Soundness and Completeness.

If α is non-trivial, then there is a vertex $v \in \Sigma^*$ such that $\alpha \cdot v \neq v$, which $\mathcal{M}$ can nondeterministically choose to write on its check-stack and thus accept α. If α is trivial, then $\alpha \cdot v = v$ for each vertex $v \in \Sigma^*$, and there is no choice of checking stack for which $\mathcal{M}$ will accept, so $\mathcal{M}$ will reject.

Thus, $\mathcal{M}$ accepts a word if and only if it is in coWP_Y. □

3.3. Open Problems and Concluding Remarks

Theorem D opens the door to a new characterisation of groups by their co-word problem. In particular, this result is a step towards a characterisation of groups with ET0L co-word problems. However, it still remains to be shown that there is a group whose co-word problem is ET0L but not context-free. It is conjectured [17, p. 2] that Grigorchuk's group does not have a context-free co-word problem. Thus, we ask the following.

Question 3.7. *Is the co-word problem for Grigorchuk's group (or some other bounded automata group) not context-free?*

It is then natural to ask if there is a classification of co-ET0L groups.

Chapter 4

Polyhedral Sets and Polyhedrally Constrained Languages

In this chapter we introduce the family of *polyhedrally constrained languages* which we use in the proof of Theorem A. This family of languages is a generalisation of linearly constrained languages, as in Definition 2.13. It is a result of Massazza [76] that the generating function of linearly constrained languages is holonomic (see Proposition 2.14). In Proposition 4.5, we show that the family of polyhedrally constrained languages also have holonomic (multivariate) generating functions.

Benson [10] introduced the concept of *polyhedral sets* to compute the volume growth of virtually abelian groups, in particular, for each virtually abelian group Benson constructed a polyhedral set whose (volume) generating function is the volume growth series of the group. In Chapter 5 we apply a similar argument to construct a *polyhedrally constrained language* whose generating function is the geodesic growth series of a virtually abelian group. We define and study the class of polyhedral sets in Section 4.1, and the family of polyhedrally constrained languages in Section 4.2.

4.1. Polyhedral Sets

A *polyhedral set*, as we see in Definition 4.1, is a subset of $\mathbb{Z}^n$ that encodes the integer solutions to finitely many systems of linear equations, inequalities and congruences. The class of polyhedral sets is closed under Boolean expressions, Cartesian products, and (inverse) mapping by integer affine transformation (see Propositions 4.2 and 4.3). The class of polyhedral sets and their closure properties are essential to our study of the language of geodesics for virtually abelian groups in Chapter 5.

Definition 4.1. *A subset $\mathcal{E} \subseteq \mathbb{Z}^m$ is called an* elementary region *if it can be expressed as*

$$\{z \in \mathbb{Z}^m \mid a \cdot z = b\}, \ \{z \in \mathbb{Z}^m \mid a \cdot z > b\} \ \text{ or } \ \{z \in \mathbb{Z}^m \mid a \cdot z \equiv b \pmod{c}\}$$

for some $a \in \mathbb{Z}^m$ and $b, c \in \mathbb{Z}$ with $c > 0$. A basic polyhedral set *is a finite intersection of elementary regions; and a* polyhedral set *is a finite disjoint union of basic polyhedral sets.*

From this definition we see that $\emptyset$ and $\mathbb{Z}^m$ are elementary regions, and that $\mathbb{N}^m$ is a basic polyhedral set. In Proposition 4.2 we see that the class of polyhedral sets is closed under Boolean expressions and Cartesian products.

Proposition 4.2 (Proposition 13.1 and Remark 13.2 in [10])**.** *The class of polyhedral subsets of $\mathbb{Z}^m$ is closed under finite union, finite intersection and set difference. Moreover, the class of polyhedral sets is closed under Cartesian product.*

A map $E\colon \mathbb{Z}^m \to \mathbb{Z}^n$ is an *integer affine transform* if it can be written as $E(v) = vA + b$ where $A \in \mathbb{Z}^{m\times n}$ is a matrix and $b \in \mathbb{Z}^n$ is a vector. In Proposition 4.3 we see that the class of polyhedral sets is closed under (inverse) mapping by integer affine transformations.

Proposition 4.3 (Propositions 13.7 and 13.8 in [10])**.** *Suppose that $\mathcal{P} \subseteq \mathbb{Z}^m$ and $\mathcal{Q} \subseteq \mathbb{Z}^n$ are polyhedral sets, and $E\colon \mathbb{Z}^m \to \mathbb{Z}^n$ is an integer affine transform. Then, $E(\mathcal{P})$ and $E^{-1}(\mathcal{Q})$ are both polyhedral sets.*

Notice that our definition of n-atoms and n-constraints in Definition 2.12 is similar to that of elementary regions and polyhedral sets, respectively, without modular arithmetic. From the closure properties in Proposition 4.2 we see that n-constraints form a subclass of the polyhedral subsets of $\mathbb{Z}^n$. It can be verified by the reader that, for each $n \geqslant 1$,

$$\{(x, 0, 0, \ldots, 0) \in \mathbb{Z}^n \mid x \equiv 0 \pmod 2\}$$

is a polyhedral set but not an n-constraint, and thus the class of n-constraints form a proper subclass of the polyhedral subsets of $\mathbb{Z}^n$. In the following section, we generalise the family of linearly constrained languages, defined in Definition 2.13, to the family of *polyhedral constrained languages* which we use in the proof of Theorem A.

4.2. Polyhedrally Constrained Languages

In Section 2.3, we saw that a constrained language, $L(U, \mathcal{C})$, is the intersection of an unambiguous context-free language $U \subseteq \Sigma$ and the set of words whose Parikh images belong to a set $\mathcal{C} \subseteq \mathbb{Z}^{|\Sigma|}$. Moreover, we defined the family of linearly constrained languages in Definition 2.13 as the constrained languages where $\mathcal{C}$ is a $|\Sigma|$-constraint. In this section we generalise this definition to the family of *polyhedrally constrained languages* as follows.

Definition 4.4. *A language is* polyhedraly constrained *if it can be written as*

$$L(U, \mathcal{P}) = \{w \in U \subseteq \Sigma^* \mid \Phi_\Sigma(w) \in \mathcal{P}\}$$

where U is an unambiguous context-free language, and $\mathcal{P}$ is a polyhedral set.

In Chapter 5 we will not require the full power of polyhedrally constrained languages, in particular, we only require polyhedrally constrained languages $L(U, \mathcal{P})$ where U is a regular language. It can then be shown that such languages form a subfamily of the *RCM languages* introduced by Castiglione and Massazza [23], moreover, they showed that the single-variable generating functions of these languages are holonomic. In Proposition 4.5, we show that the multivariate generating function of each polyhedrally constrained language is holonomic. We make use of this characterisation in the proof of Theorem A.

Proposition 4.5. *The multivariate generating function of a polyhedrally constrained language is holonomic.*

Proof. Let $L(U, \mathcal{P}) \in \Sigma^*$ be a polyhedrally constrained language. From the definition of polyhedral sets, we may decompose $\mathcal{P} \subseteq \mathbb{Z}^{|\Sigma|}$ into a union of finitely many disjoint basic polyhedral sets $\mathcal{P} = \bigcup_{i=1}^{L} \mathcal{B}_i$. Moreover, each such basic polyhedral set $\mathcal{B}_i \subseteq \mathbb{Z}^{|\Sigma|}$ can be written as a finite intersection of elementary regions

$$\mathcal{B}_i = \bigcap_{j=1}^{K_{i,1}} \{v \in \mathbb{Z}^{|\Sigma|} \mid \alpha_{i,j} \cdot v = \beta_{i,j}\} \cap \bigcap_{j=1}^{K_{i,2}} \{v \in \mathbb{Z}^{|\Sigma|} \mid \xi_{i,j} \cdot v > \lambda_{i,j}\} \cap \bigcap_{j=1}^{K_{i,3}} \{v \in \mathbb{Z}^{|\Sigma|} \mid \zeta_{i,j} \cdot v \equiv \eta_{i,j} \pmod{\theta_{i,j}}\}$$

where each $\alpha_{i,j}, \xi_{i,j}, \zeta_{i,j} \in \mathbb{Z}^{|\Sigma|}$, each $\beta_{i,j}, \lambda_{i,j}, \eta_{i,j} \in \mathbb{Z}$, and $\theta_{i,j} \in \mathbb{N}_+$.

From the definition of constrained language we see that $L(U, \mathcal{P})$ is the union of disjoint polyhedrally constrained languages $L(U, \mathcal{B}_i)$. We see that if each $L(U, \mathcal{B}_i)$ has a multivariate generating function of $f_i(x_1, x_2, \ldots, x_{|\Sigma|})$, then the multivariate generating function for $L(U, \mathcal{P})$ is given by

$$f(x_1, x_2, \ldots, x_{|\Sigma|}) = \sum_{i=1}^{L} f_i(x_1, x_2, \ldots, x_{|\Sigma|}).$$

For each basic polyhedral set $\mathcal{B}_i$, we introduce a $|\Sigma|$-constraint

$$\mathcal{C}_i = \bigcap_{j=1}^{K_{i,1}} \{v \in \mathbb{Z}^{|\Sigma|} \mid \alpha_{i,j} \cdot v = \beta_{i,j}\} \cap \bigcap_{j=1}^{K_{i,2}} \{v \in \mathbb{Z}^{|\Sigma|} \mid \xi_{i,j} \cdot v > \lambda_{i,j}\},$$

and a monoid homomorphism $\varphi_i \colon \Sigma^* \to \prod_{j=1}^{K_{i,3}} (\mathbb{Z}/\theta_{i,j}\mathbb{Z})$ such that

$$\varphi_i(w) = (\zeta_{i,1} \cdot \Phi_\Sigma(w),\ \zeta_{i,2} \cdot \Phi_\Sigma(w),\ \ldots,\ \zeta_{i,K_{i,3}} \cdot \Phi_\Sigma(w));$$

moreover, we write $R_i \in \Sigma^*$ for the inverse image

$$R_i = \varphi_i^{-1}(\{(\eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,K_{i,3}})\}).$$

Each language $R_i \in \Sigma^*$ is expressed as the inverse image of a subset of a finite monoid. From [84, Theorem 1] we see that each R_i is a regular language, in particular, for each R_i we may construct a finite-state automaton with states given by the set $\prod_{j=1}^{K_{i,3}} (\mathbb{Z}/\theta_{i,j}\mathbb{Z})$, initial state given by $(0, \ldots, 0)$, an accepting state of $(\eta_{i,1}, \eta_{i,2}, \ldots, \eta_{i,K_{i,3}})$, and a transition $v \to^\sigma v'$ for each state v and letter $\sigma \in \Sigma$ where $v' = v + \varphi_i(\sigma)$. Moreover, since the class of unambiguous context-free grammar is closed under intersection with regular language (see Theorem 6.4.1 on p. 197 of [58]), we see that each $L(U \cap R_i, \mathcal{C}_i) = L(U, \mathcal{B}_i)$ is linearly constrained as in Definition 2.13. From Proposition 2.14, we see that each $f_i(x_1, x_2, \ldots, x_{|\Sigma|})$ is holonomic.

From Lemma 2.9, holonomic functions are closed under addition, and thus the multivariate generating function of $L(U, \mathcal{P})$ is holonomic. □

Chapter 5

Virtually Abelian Groups

It is a well-known result of Gromov [56] that a group has polynomial volume growth if and only if it is virtually nilpotent. Moreover, from the work of Bass [9] we know that virtually nilpotent groups have polynomial volume of integer degrees. Bridson, Burillo, Elder and Šunić [19] asked if there is an analogous classification for groups with polynomial geodesic growth and if there exists a group with intermediate geodesic growth. Towards these questions they provided a sufficient condition, given in Lemma 5.1, for a virtually abelian group to have polynomial geodesic growth, and furnished an example of a virtually $\mathbb{Z}^2$ group, given in Equation (5.1), with polynomial geodesic growth. Before this virtually $\mathbb{Z}^2$ example, the only groups known to have polynomial geodesic growth were virtually cyclic. In this chapter, we take the next step towards a classification of polynomial geodesic growth by characterising the geodesic growth series for all virtually abelian groups with respect to any finite weighted monoid generating sets.

Lemma 5.1 (Theorem 1 in [19]). *Let G be a finitely-generated group. If there is an element $g \in G$ whose normal closure is a finite-index abelian subgroup of G, then G has polynomial geodesic growth with respect to some generating set.*

In [19], it was shown that the virtually $\mathbb{Z}^2$ group

$$\langle a, b, t \mid [a, b] = t^2 = 1,\ a^t = b \rangle \tag{5.1}$$

has polynomial geodesic growth with respect to the generating set $\{a, a^{-1}, t\}$. This group was introduced by Cannon [43, Example 4.4.1 on p. 97] as an example of a group that is *short-lex automatic* with respect to one, but not all, generating sets. Moreover, it was shown in Example 4.4.2 on page 98 of [43] that this group has a generating set for which the geodesics do not form a regular language. In Theorem D, we show that the language of geodesics for each virtually abelian group is blind multicounter for every generating set.

We may generalise the construction given in Equation (5.1) to show that for any finitely-generated abelian group, A, there is a virtually-A group with polynomial geodesic growth. Let A be a finitely-generated abelian group, then from the classification of

finitely-generated abelian groups we see that $A = F \times \mathbb{Z}^n$ for some n and finite group F. Let $x_1, x_2, \ldots, x_n$ be the standard basis for the $\mathbb{Z}^n$ subgroup of A. Let

$$B = \left\langle A, t \;\middle|\; [F,t] = 1,\ x_i^t = x_{i+1} \text{ for each } i < n,\ t^n = 1 \right\rangle. \tag{5.2}$$

We see that B contains A as a subgroup of index n; and thus B contains $\mathbb{Z}^n$ as a subgroup of index $n|F|$. Moreover, the normal closure of x_1 in B is the finite-index free-abelian subgroup $\mathbb{Z}^n$. From Lemma 5.1, we see that B has polynomial geodesic growth with respect to some generating set.

Benson [10] showed that the volume growth series for virtually abelian groups is rational with respect to any finite (weighted monoid) generating set. This result was generalised by Evetts [44] who showed that the coset, subgroup, and conjugacy growth series of a virtually abelian group is rational with respect to any finite (weighted monoid) generating set. In Sections 5.1 to 5.1.2 we modify the methods of Benson, and provide a characterisation of the geodesic growth series in Section 5.2 by combining this with our result on polyhedrally constrained language given in Proposition 4.5.

5.1. Patterned Words

Let G be a virtually abelian group that is generated as a monoid by some finite weighted generating set S. It is known that G contains a finite-index normal subgroup that is isomorphic to $\mathbb{Z}^n$ for some n. This follows as G must contain an abelian subgroup A of finite index, then from the classification of finitely generated abelian groups we see that G must contain a group H that is isomorphic to $\mathbb{Z}^n$. We may then obtain a normal subgroup from the core of H as $\bigcap_{g \in G} H^g$. This subgroup will be finite index in G, and is free abelian as it is a subgroup of a free abelian group H [86, pp. 100–1]. Without loss of generality, we assume that $\mathbb{Z}^n \triangleleft G$ with $d = [G : \mathbb{Z}^n]$. We fix a set of coset representatives $T = \{t_1 = 1, t_2, \ldots, t_d\}$ for $\mathbb{Z}^n$ in G. We then write elements of G in the normal form $g = z \cdot t$ where $z \in \mathbb{Z}^n$ and $t \in T$.

Definition 5.2. *Let $\psi \colon G \to \mathbb{Z}^n$ and $\rho \colon G \to T$ be the maps defined such that the normal form for $g \in G$ is given by $\psi(g) \cdot \rho(g)$.*

Benson [10] showed that virtually abelian groups have rational volume growth series by demonstrating that each group element has at least one geodesic representative that can be expressed as a *patterned word*, where the set of such patterned words is then studied using the theory of polyhedral sets. In this section we modify these arguments to study the set of all geodesic words in S^*, in particular, we describe Algorithm 5.15 which converts words in S^* to *patterned words* which represent the same group element with the same weight. In Section 5.1.2 we compute the weight and group element of patterned words, and describe the patterned words which correspond to geodesics.

We begin by defining two finite sets of words $Y, P \subseteq S^*$ as follows.

Definition 5.3. *From the generating set S and the normal subgroup $\mathbb{Z}^n \triangleleft G$ with finite index $d = [G : \mathbb{Z}^n]$, we define the sets*

$$Y = \{\sigma \in S^* \mid 1 \leqslant |\sigma|_S \leqslant d \text{ and } \overline{\sigma} \in \mathbb{Z}^n\} \quad \textit{and}$$
$$P = \{\sigma \in S^* \mid 1 \leqslant |\sigma|_S \leqslant d-1 \text{ and } \overline{\sigma} \notin \mathbb{Z}^n\},$$

and we fix a labelling $\{y_1, y_2, \ldots, y_m\} = Y$ where $m = |Y|$.

We define the sets Y and P as above so that we have the technical property given in Lemma 5.4. We will find this property useful in the proof of Lemma 5.14 which is then used to construct Algorithm 5.15.

Lemma 5.4. *Suppose that $w \in S^*$ with $1 \leqslant |w|_S \leqslant d$ and $w \notin P$. Then, there is a factoring $w = \alpha\beta\delta$ with $\alpha \in P \cup \{\varepsilon\}$, $\beta \in Y$ and $\delta \in S^*$. In particular, there is a unique choice of such a factoring for which $(|\alpha|_S, |\beta|_S) \in \mathbb{N}^2$ is minimal with respect to the lexicographic ordering on $\mathbb{N}^2$.*

Proof. Let $w = w_1 w_2 \cdots w_k$ with $1 \leqslant k \leqslant d$ and $w \notin P$.

Notice that if we have at least one such factorisation, then there is a unique choice of such a factoring where $(|\alpha|_S, |\beta|_S) \in \mathbb{N}^2$ is minimal with respect to the lexicographic ordering on $\mathbb{N}^2$. Thus, all that remains to be shown is that at least one such factoring $w = \alpha\beta\delta$ exists.

If $|w|_S < d$, then we have such a factorisation given by $\beta = w$, and $\alpha = \delta = \varepsilon$. Thus, in the remainder of this proof we consider the case where $|w|_S = d$.

If $|w|_S = d$, then from the pigeonhole principle on the d cosets, we see that there must be a nontrivial factor $b = w_i w_{i+1} \cdots w_j$ for which $\overline{b} \in \mathbb{Z}^n$. Let $I \geqslant 1$ be the smallest value for which there is a $J \geqslant I$ with $\overline{w_I w_{I+1} \cdots w_J} \in \mathbb{Z}^n$, then let $\alpha = w_1 w_2 \cdots w_{I-1}$ and $\beta = w_I w_{I+1} \cdots w_J$. From our choice of indices I and J, we see that $\beta \in Y$, and either $\alpha = \varepsilon$ or $\overline{\alpha} \notin \mathbb{Z}^n$. Moreover, we see that $|\alpha|_S = I - 1 \leqslant d - 1$ and thus $\alpha \in P \cup \{\varepsilon\}$. □

Notice that $S \subseteq Y \cup P$, and thus $Y \cup P$ generates the group G. We will see that for each word $\sigma \in S^*$, there is a word $w \in Y^*(PY^*)^k$, with $0 \leqslant k \leqslant d$, such that w represents the same group element as σ with the same weight. We formalise this by defining *patterns* and *patterned words* as follows.

Definition 5.5 (Patterned words)**.** *Let $\pi = \pi_1 \pi_2 \cdots \pi_k \in P^*$ be a word in the letters of P with length $k = |\pi|_P \leqslant d$ for which each proper prefix belongs to a distinct coset, that is,*

$$1 = \rho(\overline{\varepsilon}),\ \rho(\overline{\pi_1}),\ \rho(\overline{\pi_1\pi_2}),\ \ldots,\ \rho(\overline{\pi_1\pi_2\cdots\pi_{k-1}}) \tag{5.3}$$

are pairwise distinct; and let $v \in \mathbb{N}^{(k+1)m}$ be a vector where $m = |Y|$. Then we say that π is a pattern *and that (v, π) is a* patterned word*. We then write*

$$v^\pi = \left(y_1^{v_1} y_2^{v_2} \cdots y_m^{v_m}\right)\pi_1\left(y_1^{v_{m+1}} y_2^{v_{m+2}} \cdots y_m^{v_{2m}}\right)\pi_2 \cdots \pi_k\left(y_1^{v_{k\cdot m+1}} y_2^{v_{k\cdot m+2}} \cdots y_m^{v_{(k+1)\cdot m}}\right).$$

Notice that $\rho(\overline{\pi})$ is not included in (5.3). If $\rho(\overline{\pi})$ is also distinct from each coset representative in (5.3), then we say that π is a strong pattern *and that (v, π) is a* strongly patterned word*.*

To simplify notation in later sections, we introduce the following sets.

Definition 5.6. *We write* $\textsc{Patt} \subseteq P^*$ *for the set of all patterns, and we write* $\textsc{StrPatt} \subseteq \textsc{Patt}$ *for the set of all strong patterns. Notice that* $\textsc{Patt}$ *and* $\textsc{StrPatt}$ *are finite, in particular,* $|\textsc{Patt}| \leqslant |P|^{d+1}$.

To simplify notation in Section 5.1.1, we extend this as follows.

Definition 5.7 (Extended Patterned Words)**.** *If* (v, π) *is a (strongly) patterned word, and* $\sigma \in S^*$*, then* $((v, \pi), \sigma)$ *is an* extended (strongly) patterned word.

In Algorithm 5.15, for each word $\sigma \in S^*$, we construct a finite sequence of extended patterned words that begins with $((\mathbf{0}, \varepsilon), \sigma)$ and ends with an extended patterned word of the form $((v, \pi), \varepsilon)$. Moreover, this sequence has the property that v^π and σ represent the same group element with the same weight. To simplify notation, we define the following equivalence relation.

Definition 5.8. *We define the equivalence relation* $\simeq$ *on* S^* *such that, for each* $w, \sigma \in S^*$*, we have* $w \simeq \sigma$ *if and only if both* $\overline{w} = \overline{\sigma}$ *and* $\omega(w) = \omega(\sigma)$.

Notice that if we have a patterned word (v, π) with $v^\pi \simeq \sigma$, then σ is a geodesic if and only if the word v^π is a geodesic.

5.1.1. Word Shuffling

In this section we construct Algorithm 5.15 which 'shuffles' words of the form $\sigma \in S^*$ into patterned words (v, π) which represent the same group element with the same weighted length. In particular, for each word σ, we compute a finite sequence of extended patterned words

$$((\mathbf{0}, \varepsilon), \sigma) = ((u^{(1)}, \tau^{(1)}), \sigma^{(1)}), ((u^{(2)}, \tau^{(2)}), \sigma^{(2)}), \\ \ldots, ((u^{(q)}, \tau^{(q)}), \sigma^{(q)}) = ((v, \pi), \varepsilon), \quad (5.4)$$

such that

$$(u^{(i)})^{\tau^{(i)}} \sigma^{(i)} \simeq (u^{(i+1)})^{\tau^{(i+1)}} \sigma^{(i+1)} \quad \text{and} \quad |\sigma^{(i)}|_S > |\sigma^{(i+1)}|_S$$

for each i. Notice that $v^\pi \simeq \sigma$, and $q \leqslant |\sigma|_S + 1$ where q is the length of the sequence in (5.4). From (5.4), we define $\text{Shuffle}(\sigma) = (v, \pi)$ where the patterned word (v, π) has the property that $v^\pi \simeq \sigma$.

The idea of Algorithm 5.15 is to compute each $((u^{(i+1)}, \tau^{(i+1)}), \sigma^{(i+1)})$ from its previous extended patterned word $((u^{(i)}, \tau^{(i)}), \sigma^{(i)})$ by replacing a bounded-length prefix of $\sigma^{(i)}$ with a strictly shorter word, adding at most a unit vector to $u^{(i)}$, and adding at most one letter to $\tau^{(i)}$. In order to describe our algorithm, we introduce the following additional notation.

Recall that $d = [G : \mathbb{Z}^n]$ is the index of the $\mathbb{Z}^n$ normal subgroup of G. For each word $\sigma \in S^*$, we fix a bounded-length prefix as follows.

Definition 5.9 (Prefixes)**.** *We write* $\mathrm{Prefix}\colon S^* \to S^*$ *for the function which computes the prefix of a word of length at most* d*, that is,* $\mathrm{Prefix}(\sigma) = \sigma_1\sigma_2\cdots\sigma_q$ *where* $q = \min(d, |\sigma|_S)$*. Notice that if* $w = \mathrm{Prefix}(\sigma)$ *with* $|w|_S < d$*, then* $\sigma = w$*.*

In sequence (5.4), each word $\sigma^{(i+1)}$ is obtained from $\sigma^{(i)}$ by replacing the prefix $w^{(i)} = \mathrm{Prefix}(\sigma^{(i)})$ with a strictly shorter word $w^{(i)\prime}$. We write these prefix replacements using the following notation.

Definition 5.10 (Prefix Replacements)**.** *Let* $\sigma \in S^*$ *be a word which factors as* $\sigma = w\zeta$ *where* $w, \zeta \in S^*$*, then for each word* $w' \in S^*$ *we write* $(w \mapsto w') \cdot \sigma = w'\zeta$ *which we call a* prefix replacement*. We write a sequence of replacements as*

$$(w_n \mapsto w'_n)\cdots(w_2 \mapsto w'_2)(w_1 \mapsto w'_1)\cdot\sigma$$

where replacements are composed right-to-left. Notice that if $\sigma' = (w \mapsto w')\cdot\sigma$*, then* $\omega(\sigma') = \omega(\sigma) - \omega(w) + \omega(w')$ *where* $\omega\colon S^* \to \mathbb{N}$ *is the weight function.*

To understand how prefix replacements are composed, consider the following.

Example 5.11. *We have the sequence of replacements*

$$(c \mapsto dc)(ba \mapsto cb)(\varepsilon \mapsto b)\cdot az = dcbz. \tag{5.5}$$

Notice that if $(w \mapsto w')\cdot\sigma$ *is defined, then we have* $\sigma = (w' \mapsto w)(w \mapsto w')\cdot\sigma$*, that is, each prefix replacement has an inverse. For example, from the sequence of prefix replacements given in (5.5), we see that*

$$az = (b \mapsto \varepsilon)(cb \mapsto ba)(dc \mapsto c)\cdot dcbz.$$

Thus, we may compute the inverse of a sequence of prefix replacements.

For each pattern π, we write $\mathcal{N}_\pi$ for the set of all vectors v for which (v, π) is a patterned word, as defined in Definition 5.5. We introduce the following notation to simplify the description of our algorithm.

Definition 5.12. *For each pattern* $\pi = \pi_1\pi_2\cdots\pi_k \in P^*$*, we write* $\mathcal{Z}_\pi$ *and* $\mathcal{N}_\pi$ *for the sets* $\mathbb{Z}^{(k+1)m}$ *and* $\mathbb{N}^{(k+1)m}$*, respectively, where* $m = |Y|$*. Moreover, for each* $i \in \{1, 2, \ldots, \dim(\mathcal{Z}_\pi)\}$ *we write* $e_{\pi,i}$ *for the* i*-th standard basis element of* $\mathcal{Z}_\pi$ *and* $e_{\pi,\varnothing} = \mathbf{0} \in \mathcal{Z}_\pi$ *for the zero vector of* $\mathcal{Z}_\pi$*.*

When computing (5.4), it may be the case that $|\tau^{(i)}|_P \neq |\tau^{(i+1)}|_P$ and thus the vectors $u^{(i)}$ and $u^{(i+1)}$ lie in different spaces $\mathcal{N}_{\tau^{(i)}}$ and $\mathcal{N}_{\tau^{(i+1)}}$, respectively. We define the following map to convert between these spaces.

Definition 5.13. *For each pair of patterns* $\pi, \tau \in P^*$*, let* $t = \dim(\mathcal{Z}_\tau)$ *and* $p = \dim(\mathcal{Z}_\pi)$*, then we define the map* $\mathrm{Proj}_{\pi,\tau}\colon \mathcal{Z}_\pi \to \mathcal{Z}_\tau$ *such that*

$$\mathrm{Proj}_{\pi,\tau}(u_1, u_2, \ldots, u_p) = (u_1, u_2, \ldots, u_p, 0, 0, \ldots, 0)$$

if $t > p$*, and*

$$\mathrm{Proj}_{\pi,\tau}(u_1, u_2, \ldots, u_p) = (u_1, u_2, \ldots, u_t)$$

otherwise. Notice that if $\dim(\mathcal{Z}_\tau) < \dim(\mathcal{Z}_\pi)$*, then* $\mathrm{Proj}_{\pi,\tau}$ *is a projection; otherwise,* $\dim(\mathcal{Z}_\tau) \geqslant \dim(\mathcal{Z}_\pi)$ *and* $\mathrm{Proj}_{\pi,\tau}$ *is an embedding.*

In order to construct Algorithm 5.15, we need to define a map which explicitly describes how to construct the sequence of extended patterned words in (5.4). We construct such a map in the following lemma.

Lemma 5.14. *We may construct a map*

$$\Delta\colon \mathit{StrPAtt} \times W_1 \to (\mathbb{N}_+ \cup \{\varnothing\}) \times \mathit{PAtt} \times W_2,$$

where

$$W_1 = \{w \in S^* \mid 1 \leqslant |w|_S \leqslant d\} \quad \text{and} \quad W_2 = \{w \in S^* \mid |w|_S < d\}$$

with the following properties. Let $((u,\tau),\sigma)$ *be an extended strongly patterned word, and let* $\Delta(\tau, w) = (x, \tau', w')$ *with* $w = \mathrm{Prefix}(\sigma)$*. We may then* apply Δ *to obtain an extended patterned word* $((u',\tau'),\sigma')$ *where* $u' = \mathrm{Proj}_{\tau,\tau'}(u) + e_{\tau',x}$ *and* $\sigma' = (w \mapsto w') \cdot \sigma$*. This will be denoted as*

$$((u,\tau),\sigma) \xrightarrow{\Delta} ((u',\tau'),\sigma').$$

For each extended strongly patterned word $((u,\tau),\sigma)$*,*

1. $|\tau|_P \leqslant |\tau'|_P$ *and thus* $\mathrm{Proj}_{\tau,\tau'}\colon \mathcal{N}_\tau \to \mathcal{N}_{\tau'}$ *is an embedding;*
2. $u^\tau \sigma \simeq (u')^{\tau'} \sigma'$*;*
3. $|\sigma|_S > |\sigma'|_S$ *and* $\omega(w) > \omega(w')$*; and*
4. *either* $|\sigma'|_S = 0$*, or* $((u',\tau'),\sigma')$ *is an extended strongly patterned word.*

Notice that property 4 implies that either $((u',\tau'),\sigma')$ *is equivalent to the patterned word* (u',τ')*, or we may apply* Δ *again. From property 3, we see that after finitely many applications of the map* Δ*, we have a patterned word.*

Proof. Let $\tau = \tau_1\tau_2\cdots\tau_k \in P^*$ be a strong pattern, that is, τ is a pattern for which the coset representatives

$$\rho(\overline{\varepsilon}),\ \rho(\overline{\tau_1}),\ \rho(\overline{\tau_1\tau_2}),\ \rho(\overline{\tau_1\tau_2\tau_3}),\ \ldots,\ \rho(\overline{\tau})$$

are pairwise distinct. Then, from the pigeonhole principle on the d cosets of $\mathbb{Z}^n$ in G, we see that $|\tau|_P = k < d$.

Let $w \in S^*$ be a word with length $1 \leqslant |w|_S \leqslant d$. We separate the remainder of this proof into the cases where $w \in P$ and $w \notin P$ as follows.

Suppose that $w \in P$, then we have a length $k+1$ pattern $\tau' = \tau w$, moreover, from the definition of words in P, we see that $|w|_S < d$, and from Definition 5.9 we have $w = \sigma$. We then define $\Delta(\tau, w) = (\varnothing, \tau', \varepsilon)$. For each extended strongly patterned word $((u,\tau),w)$ with $u \in \mathbb{N}^p$, we then obtain an extended patterned word $((u',\tau'),\varepsilon)$ where $u' = \mathrm{Proj}_{\tau,\tau'}(u) = (u_1, u_2, \ldots, u_p, 0, 0, \ldots, 0)$. Notice that we have $(u')^{\tau'} = u^\tau w$. This completes our proof for the case that $w \in P$.

In the remainder of this proof, we suppose that $w \notin P$. From Lemma 5.4, we factor w uniquely as $w = \alpha\beta\delta$ where $\alpha \in P \cup \{\varepsilon\}$, $\beta \in Y$ and $(|\alpha|_S, |\beta|_S)$ is minimal with respect to the lexicographic order on $\mathbb{N}^2$. From the labelling $Y = \{y_1, y_2, \ldots, y_m\}$, we see that there must be an index b such that $\beta = y_b$.

Let $((u,\tau),\sigma)$ be an extended strongly patterned word with $w = \mathrm{Prefix}(\sigma)$ and $u = (u_0, u_1, \ldots, u_k)$ where each $u_a = (u_{a,1}, u_{a,2}, \ldots, u_{a,m}) \in \mathbb{N}^m$. Then if we factor σ as $\sigma = w\zeta$, we see that

$$u^\tau\sigma = \left(y_1^{u_{0,1}} y_2^{u_{0,2}} \cdots y_m^{u_{0,m}}\right)\tau_1\left(y_1^{u_{1,1}} y_2^{u_{1,2}} \cdots y_m^{u_{1,m}}\right)\tau_2 \\ \cdots \tau_{k-1}\left(y_1^{u_{k-1,1}} y_2^{u_{k-1,2}} \cdots y_m^{u_{k-1,m}}\right)\tau_k\left(y_1^{u_{k,1}} y_2^{u_{k,2}} \cdots y_m^{u_{k,m}}\right)\alpha y_b\delta\zeta.$$

If there is an index a with $0 \leqslant a \leqslant k$ such that $\rho(\overline{\tau_1\tau_2\cdots\tau_a}) = \rho(\overline{\pi\alpha})$, then the choice of such an index a must be unique, and we see that

$$\overline{\tau_{a+1}\left(y_1^{u_{a+1,1}} y_2^{u_{a+1,2}} \cdots y_m^{u_{a+1,m}}\right)\tau_{a+2}\cdots\tau_k\left(y_1^{u_{k,1}} y_2^{u_{k,2}} \cdots y_m^{u_{k,m}}\right)\alpha} \in \mathbb{Z}^n$$

commutes with $\overline{y_b} \in \mathbb{Z}^n$, that is,

$$(u_0, \ldots, u_{a-1}, u_a + e_b, u_{a+1}, \ldots, u_k)^\tau \alpha\delta\zeta \simeq u^\tau \alpha y_b \delta\zeta = u^\tau\sigma$$

where $e_b \in \mathbb{N}^m$ is the b-th standard basis element. In this case we define the map $\Delta(\tau, w) = (a \cdot m + b,\ \tau,\ \alpha\delta)$ and our proof is complete. Otherwise, we see that the coset representatives

$$\rho(\overline{\varepsilon}),\ \rho(\overline{\tau_1}),\ \rho(\overline{\tau_1\tau_2}),\ \rho(\overline{\tau_1\tau_2\tau_3}),\ \ldots,\ \rho(\overline{\tau}),\ \rho(\overline{\tau\alpha})$$

are pairwise distinct and $\alpha \neq \varepsilon$, that is, $\alpha \in P$. Then we see that the length $k+1$ word $\tau' = \tau\alpha \in P^*$ is a strong pattern, and that we have

$$(u_0, u_2, \ldots, u_k, e_b)^{\tau'}\delta\zeta = u^\tau \alpha y_b \delta\zeta = u^\tau\sigma$$

where $e_b \in \mathbb{N}^m$ is the b-th standard basis vector. After defining $\Delta(\tau, w) = (a \cdot k + b, \tau', \delta)$ our proof is complete. □

We are now ready to define our algorithm as follows.

Algorithm 5.15 (Word Shuffling)**.** *Let Δ be the map in Lemma 5.14. For each word $\sigma \in S^*$, there is a finite sequence of extended patterned words*

$$((\mathbf{0},\varepsilon),\sigma) = ((u^{(1)},\tau^{(1)}),\sigma^{(1)}) \xrightarrow{\Delta} ((u^{(2)},\tau^{(2)}),\sigma^{(2)}) \\ \cdots \xrightarrow{\Delta} ((u^{(q)},\tau^{(q)}),\sigma^{(q)}) = ((v,\pi),\varepsilon). \quad (5.6)$$

From this sequence we define $\mathrm{Shuffle}(\sigma) = (v,\pi)$*. Notice from property 3 in Lemma 5.14 that we have*

$$|\sigma|_S = |\sigma^{(1)}|_S > |\sigma^{(2)}|_S > |\sigma^{(3)}|_S > \cdots > |\sigma^{(q)}|_S = 0$$

and thus $q \leqslant |\sigma|_S + 1$. From property 2 in Lemma 5.14, we see that $v^\pi \simeq \sigma$.

In the remainder of this section, we compute the group elements and weights of patterned words, and determine which patterned word represents geodesics.

5.1.2. Geodesic Patterned Words

From Algorithm 5.15, for each word $\sigma \in S^*$ we have a well-defined patterned word $(v, \pi) = \text{Shuffle}(\sigma)$ such that $v^\pi \simeq \sigma$, that is, v^π represents the same group element as σ with the same weight. We see that σ is a geodesic if and only if v^π is a geodesic.

In this section, we modify an argument of Benson [10] and show that the group element and weight of any word v^π can be computed with the use of integer affine transforms, and that we may verify that v^π is a geodesic by checking if the vector v belongs to a polyhedral set $\mathcal{G}_\pi$.

Lemma 5.16. *For each pattern π, there are integer affine transformations $\Psi_\pi \colon \mathcal{Z}_\pi \to \mathbb{Z}^n$ and $\Omega_\pi \colon \mathcal{Z}_\pi \to \mathbb{Z}$ such that for each patterned word (v, π), we have $\overline{v^\pi} = \Psi_\pi(v) \cdot \rho(\overline{\pi})$ and $\omega(v^\pi) = \Omega_\pi(v)$.*

Proof. Recall that in Definition 5.3 we fixed a labelling $Y = \{y_1, y_2, \ldots, y_m\}$ where $m = |Y|$. Define the matrix $Z \in \mathbb{Z}^{m \times n}$ such that $e_i Z = \overline{y_i}$ for each standard basis vector $e_i \in \mathbb{Z}^m$. Then, we see that $vZ = \overline{y_1^{v_1} y_2^{v_2} \cdots y_m^{v_m}}$ for each $v \in \mathbb{N}^m$. For each $p \in P$ we see that $\overline{p} x \overline{p}^{-1} \in \mathbb{Z}^n$ for each $x \in \mathbb{Z}^n \triangleleft G$; thus we define matrices $R_p \in \mathbb{Z}^{n \times n}$ such that $x R_p = \overline{p} x \overline{p}^{-1}$ for each $x \in \mathbb{Z}^n$.

To compute the element $\overline{v^\pi}$ we first rewrite v^π as

$$\begin{aligned}\left(y_1^{v_1} y_2^{v_2} \cdots y_m^{v_m}\right) \cdot \pi_1 \left(y_1^{v_{m+1}} y_2^{v_{m+2}} \cdots y_m^{v_{2m}}\right) \pi_1^{-1} \\ (\pi_1 \pi_2) \left(y_1^{v_{2m+1}} y_2^{v_{2m+2}} \cdots y_m^{v_{3m}}\right) (\pi_1 \pi_2)^{-1} \\ \cdots \pi \left(y_1^{v_{km+1}} y_2^{v_{km+2}} \cdots y_m^{v_{(k+1)m}}\right) \pi^{-1} \cdot \pi.\end{aligned}$$

Then we see that $\rho(\overline{v^\pi}) = \rho(\overline{\pi})$ and $\psi(\overline{v^\pi}) = \Psi_\pi(v)$ where

$$\begin{aligned}\Psi_\pi(v) = (v_1, v_2, \ldots, v_m) Z + (v_{m+1}, v_{m+2}, \ldots, v_{2m}) Z R_{\pi_1} + \\ \cdots + (v_{mk+1}, v_{mk+2}, \ldots, v_{(k+1)m}) Z R_{\pi_k} \cdots R_{\pi_2} R_{\pi_1} + \psi(\pi).\end{aligned}$$

Considering the word v^π we see that $\omega(v^\pi) = \Omega_\pi(v)$ where

$$\Omega_\pi(v) = \omega(\pi) + \sum_{j=0}^{k} \sum_{i=1}^{m} v_{jm+i} \cdot \omega(y_i).$$

The maps $\Psi_\pi \colon \mathcal{Z}_\pi \to \mathbb{Z}^n$ and $\Omega_\pi \colon \mathcal{Z}_\pi \to \mathbb{Z}$ are integer affine transforms. □

From the integer affine transformations defined in Lemma 5.16 and the closure properties of polyhedral sets we have the following result.

Lemma 5.17. *For each pattern π, there is a polyhedral set $\mathcal{G}_\pi \subseteq \mathcal{N}_\pi$ such that $v \in \mathcal{G}_\pi$ if and only if (v, π) is a patterned word where v^π is a geodesic.*

Proof. From Algorithm 5.15 we see that the word v^π is a geodesic if and only if there is no patterned word (u, τ) with $\overline{u^\tau} = \overline{v^\pi}$ and $\omega(u^\tau) < \omega(v^\pi)$. For each pattern π, let $E_\pi \colon \mathcal{Z}_\pi \to \mathbb{Z}^{n+1}$ be the integer affine transformation defined as $E_\pi(v) = (\Psi_\pi(v), \Omega_\pi(v))$, and let $\mathcal{R} \subseteq \mathbb{Z}^{2(n+1)}$ be the polyhedral set

$$\mathcal{R} = \left\{ (\nu, \mu) \in \mathbb{Z}^{n+1} \times \mathbb{Z}^{n+1} \;\middle|\; \begin{array}{r} \nu_1 = \mu_1,\, \nu_2 = \mu_2,\, \ldots,\, \nu_n = \mu_n \\ \text{and } \nu_{n+1} > \mu_{n+1} \end{array} \right\}.$$

Then, we see that v^π is geodesic if and only if there is no patterned word (u, τ) with $\rho(\overline{\tau}) = \rho(\overline{\pi})$ and $\big(E_\pi(v), E_\tau(u)\big) \in \mathcal{R}$; or equivalently, v^π is a geodesic if and only if the intersection

$$\Big(E_\pi(\{v\}) \times E_\tau(\mathcal{N}_\tau)\Big) \cap \mathcal{R}$$

is empty for each pattern τ with $\rho(\overline{\tau}) = \rho(\overline{\pi})$.

Let $f \colon \mathbb{Z}^{n+1} \times \mathbb{Z}^{n+1} \to \mathbb{Z}^{n+1}$ be the projection onto the first $\mathbb{Z}^{n+1}$ factor, that is, $f(\nu, \mu) = \nu$ for each $(\nu, \mu) \in \mathbb{Z}^{n+1} \times \mathbb{Z}^{n+1}$. Let

$$\mathcal{D}_{\pi,\tau} = \mathcal{N}_\pi \cap \Big[(E_\pi)^{-1} f\Big(\Big(E_\pi(\mathcal{N}_\pi) \times E_\tau(\mathcal{N}_\tau)\Big) \cap \mathcal{R}\Big)\Big].$$

Then, we see that v^π is a geodesic if and only if $v \notin \mathcal{D}_{\pi,\tau}$ for each pattern τ with $\rho(\overline{\tau}) = \rho(\overline{\pi})$. Then, v^π is a geodesic if and only if $v \in \mathcal{G}_\pi$ where

$$\mathcal{G}_\pi = \mathcal{N}_\pi \setminus \bigcup \Big\{ \mathcal{D}_{\pi,\tau} \;\Big|\; \tau \text{ is a pattern with } \rho(\overline{\tau}) = \rho(\overline{\pi}) \Big\}$$

where we see that the above union is finite as there can be only finitely many patterns. Moreover, from the closure properties in Propositions 4.2 and 4.3 we see that each set $\mathcal{G}_\pi \subseteq \mathcal{N}_\pi$ is polyhedral. □

5.2. Geodesic Growth

In this section we provide a characterisation of the geodesic growth of virtually abelian groups, in particular, we show that the geodesic growth of a virtually abelian group with respect to any finite weighted monoid generating set is either polynomial with rational geodesic growth series, or exponential with holonomic geodesic growth series. This result is provided in Theorem A.

The result in Theorem A is proven by first showing that the geodesic growth series is holonomic, then applying the following lemma.

Lemma 5.18. *If G has a holonomic geodesic growth series with respect to the weighted monoid generating set S, then the geodesic growth is either exponential, or the polynomial of an integer degree with a rational geodesic growth series.*

Proof. The proof follows from Corollary 2.7.1 and Lemma 2.10. □

In Lemma 5.22, we construct a weight-preserving bijection from the words in S^* to a subset of paths in a weighted graph Γ. Then, in Theorem A we construct a weight-preserving bijection from the set of such paths which correspond to geodesics, and the set of words in a finite number of polyhedrally constrained languages. Using the theory developed in Section 2.2.3, we then prove our result. We begin by constructing the graph Γ as follows.

Definition 5.19. *Let Δ be the map constructed in Lemma 5.14. Let Γ be the finite weighted directed edge-labelled graph defined as follows. For each pattern τ and each word $w \in S^*$ with $|w|_S \leqslant d = [G : \mathbb{Z}^n]$, the graph Γ has a vertex $[\tau, w] \in \mathrm{V}(\Gamma)$. Suppose $\tau \in \textsc{StrPatt}$, $|w|_S \geqslant 1$ and that $\Delta(\tau, w) = (x, \tau', w')$; if $|w|_S = d$, then for each word $\xi \in S^*$ with $|w'\xi|_S \leqslant d$, the graph Γ has a labelled edge $[\tau, w] \xrightarrow{x} [\tau', w'\xi]$; otherwise, $1 \leqslant |w|_S < d$ and the graph Γ has a labelled edge $[\tau, w] \xrightarrow{x} [\tau', w']$. Moreover, each such edge has weight $\omega(w) - \omega(w') > 0$.*

We are interested in paths of the following form.

Definition 5.20. *For each pattern π, we write $\textsc{Path}_\pi$ for the set of paths*

$$\textsc{Path}_\pi = \{p \colon [\varepsilon, w] \to^* [\pi, \varepsilon] \mid w \in S^* \text{ with } |w|_S \leqslant d\}.$$

We write $\textsc{Path}$ for the union of all such sets, that is, $\textsc{Path} = \bigcup_\pi \textsc{Path}_\pi$.

We count the instances of each edge label as follows.

Definition 5.21. *Let $\alpha \colon \textsc{Path} \to \bigcup\{\mathcal{N}_\pi \mid \pi \in \textsc{Patt}\}$ map paths $p \in \textsc{Path}_\pi$ to vectors in $\mathcal{N}_\pi$ such that the i-th component of $\alpha(p)$ counts the number of edges of p that are labelled with i, that is, if $p \colon \nu_1 \xrightarrow{x_1} \nu_2 \xrightarrow{x_2} \cdots \xrightarrow{x_k} [\pi, \varepsilon] \in \textsc{Path}_\pi$, then we have $\alpha(p) = v \in \mathcal{N}_\pi$ where each $v_i = \#\{j \mid x_j = i\}$.*

In the following lemma, we construct a weight-preserving bijection that maps from the set of words S^* to the set of paths $\textsc{Path}$.

Lemma 5.22. *We may construct a weight-preserving bijection $S^* \to \textsc{Path}$, which we denote as $\sigma \mapsto p_\sigma$, with the following properties. For each word $\sigma \in S^*$ where $\mathrm{Shuffle}(\sigma) = (v, \pi)$, we have $p_\sigma \in \textsc{Path}_\pi$. For each path $p \in \textsc{Path}_\pi$, there is a unique word $\sigma \in S^*$ such that $p = p_\sigma$ and $v^\pi \simeq \sigma$ where $v = \alpha(p) \in \mathcal{N}_\pi$.*

Proof. Let $\sigma \in S^*$, then from Algorithm 5.15 there is a finite sequence

$$\begin{aligned}((\mathbf{0}, \varepsilon), \sigma) = ((u^{(1)}, \tau^{(1)}), \sigma^{(1)}) &\xrightarrow{\Delta} ((u^{(2)}, \tau^{(2)}), \sigma^{(2)}) \\ &\cdots \xrightarrow{\Delta} ((u^{(q)}, \tau^{(q)}), \sigma^{(q)}) = ((v, \pi), \varepsilon). \quad (5.7)\end{aligned}$$

Let $w^{(j)} = \mathrm{Prefix}(\sigma^{(j)})$ and $\Delta(\tau^{(j)}, w^{(j)}) = (x^{(j+1)}, \tau^{(j+1)}, w^{(j)\prime})$.

From Lemma 5.14 we see that each $\sigma^{(j+1)} = (w^{(j)} \mapsto w^{(j)\prime}) \cdot \sigma^{(j)}$. Then,

$$\sigma = (w^{(1)\prime} \mapsto w^{(1)})(w^{(2)\prime} \mapsto w^{(2)}) \cdots (w^{(q-1)\prime} \mapsto w^{(q-1)}) \cdot \varepsilon \quad (5.8)$$

and thus

$$\omega(\sigma) = (\omega(w^{(1)}) - \omega(w^{(1)\prime})) + (\omega(w^{(2)}) - \omega(w^{(2)\prime})) \\ + \cdots + (\omega(w^{(q-1)}) - \omega(w^{(q-1)\prime})). \quad (5.9)$$

Moreover, from the properties of Δ given in Lemma 5.14, and the definition of the graph Γ given in Definition 5.19, we see that

$$p_\sigma \colon [\varepsilon, w] = [\tau^{(1)}, w^{(1)}] \xrightarrow{x^{(2)}} [\tau^{(2)}, w^{(2)}] \xrightarrow{x^{(3)}} \\ \cdots \xrightarrow{x^{(q-1)}} [\tau^{(q-1)}, w^{(q-1)}] \xrightarrow{x^{(q)}} [\tau^{(q)}, w^{(q)}] = [\pi, \varepsilon] \quad (5.10)$$

is a path in $\textsc{Path}_\pi$. Notice that the weight of the path p_σ is the same as the weight of σ in (5.9) and thus the map $\sigma \mapsto p_\sigma$ is weight preserving. It remains to be shown that the map $\sigma \mapsto p_\sigma$ is a bijection.

Suppose that we are given a path p_σ as in (5.10). Then, we may recover the words $w^{(i)\prime}$ as $\Delta(\tau^{(j)}, w^{(j)}) = (x^{(j+1)}, \tau^{(j+1)}, w^{(j)\prime})$. Hence, we may recover the word σ using equation (5.8). Thus, we see that the map $\sigma \mapsto p_\sigma$ is one-to-one. It remains to be shown that the map $\sigma \mapsto p_\sigma$ is onto, that is, for each $p \in \textsc{Path}$, there is a word σ such that $p = p_\sigma$.

Let $p \in \textsc{Path}_\pi$ be a path written as

$$p \colon [\varepsilon, w] = [\tau^{(1)}, w^{(1)}] \xrightarrow{x^{(2)}} [\tau^{(2)}, w^{(2)}] \xrightarrow{x^{(3)}} \\ \cdots \xrightarrow{x^{(q-1)}} [\tau^{(q-1)}, w^{(q-1)}] \xrightarrow{x^{(q)}} [\tau^{(q)}, w^{(q)}] = [\pi, \varepsilon].$$

Let $\Delta(\tau^{(j)}, w^{(j)}) = (x^{(j+1)}, \tau^{(j+1)}, w^{(j)\prime})$. We define the words $\sigma^{(j)}$ such that

$$\sigma^{(j)} = (w^{(j)\prime} \mapsto w^{(j)}) \cdot \sigma^{(j+1)}$$

and $\sigma^{(q)} = \varepsilon$. We define the vectors $u^{(j)} \in \mathcal{N}_{\tau^{(j)}}$ such that

$$u^{(j+1)} = \mathrm{Proj}_{\tau^{(j)}, \tau^{(j+1)}}(u^{(j)}) + e_{\tau^{(j+1)}, x^{(j+1)}}$$

and $u^{(1)} = \mathbf{0} \in \mathcal{N}_{\tau^{(1)}}$. From the property 1 in Lemma 5.14 we see that each $|\tau^{(j)}|_P \leqslant |\tau^{(j+1)}|_P$, and thus from Definition 5.13 we see that $u^{(q)} = \alpha(p)$. Let $\sigma = \sigma^{(1)}$ and $v = u^{(q)}$, then we see that

$$((\mathbf{0}, \varepsilon), \sigma) = ((u^{(1)}, \tau^{(1)}), \sigma^{(1)}) \xrightarrow{\Delta} ((u^{(2)}, \tau^{(2)}), \sigma^{(2)}) \\ \cdots \xrightarrow{\Delta} ((u^{(q)}, \tau^{(q)}), \sigma^{(q)}) = ((v, \pi), \varepsilon).$$

From this, we see that $\mathrm{Shuffle}(\sigma) = (v, \pi)$ and that $p = p_\sigma$ with $\sigma \simeq v^\pi$ where $v = \alpha(p) \in \mathcal{N}_\pi$. Moreover, we see that the map $\sigma \mapsto p_\sigma$ is onto. □

We may now prove our first main theorem as follows.

Theorem A. *Let G be a virtually abelian group with a finite weighted monoid generating set S. Then the geodesic growth with respect to S is either polynomial of integer degree with rational geodesic growth series, or exponential with holonomic geodesic growth series.*

Proof. From Lemma 5.22, we may compute the geodesic growth function as

$$\gamma_S(k) = \sum_{\pi \in \textsc{Patt}} \#\{p \in \textsc{Path}_\pi \mid \omega(p) \leqslant k \text{ and } \alpha(p) \in \mathcal{G}_\pi\} \tag{5.11}$$

where $\omega(p)$ is the weight of p, and $\mathcal{G}_\pi$ is the polyhedral set in Lemma 5.17. Notice that (5.11) is a finite sum as we have only finitely many patterns.

Let Σ be the weighted finite alphabet which contains the edges of Γ, that is, for each edge $e\colon \nu_1 \xrightarrow{x} \nu_2$ in Γ, there is a letter $(\nu_1, x, \nu_2) \in \Sigma$ with weight $\omega(e)$. Then, for each pattern π, we have a weight-preserving bijection from the paths in $\textsc{Path}_\pi$ to the words in a language $L_\pi \subseteq \Sigma^*$, in particular, the language L_π contains all words of the form

$$([\varepsilon, w], x_1, \nu_1)(\nu_1, x_2, \nu_2)(\nu_2, x_3, \nu_3)\cdots(\nu_k, x_{k+1}, [\pi, \varepsilon]) \in \Sigma^*.$$

Notice that each L_π is a regular language.

We write $\Phi(\nu_1, x, \nu_2) \in \mathbb{N}^{|\Sigma|}$ to denote the Parikh vector corresponding to the letter $(\nu_1, x, \nu_2) \in \Sigma$. For each pattern π, we define an integer affine transform $E_\pi\colon \mathbb{Z}^{|\Sigma|} \to \mathcal{Z}_\pi$ such that $E_\pi(\Phi(\nu_1, x, \nu_2)) = e_{\pi,x}$ is the x-th standard basis element for each $x \in \{1, 2, \ldots, \dim(\mathcal{Z}_\pi)\}$, and $E_\pi(\Phi(\nu_1, x, \nu_2)) = \mathbf{0}$ otherwise. Let $w \in L_\pi$ be the word corresponding to the path $p \in \textsc{Path}_\pi$, then we see that $\alpha(p) = E_\pi(\Phi(w))$. From Lemma 5.22, we see that the path p corresponds to a geodesic if and only if $\Phi(w) \in E_\pi^{-1}(\mathcal{G}_\pi)$.

For each pattern π, we define the constrained language $L_\pi^{\text{geod}} \subseteq L_\pi$ as

$$L_\pi^{\text{geod}} = \{w \in L_\pi \mid \Phi(w) \in E^{-1}(\mathcal{G}_\pi)\}.$$

Notice that there is a weight-preserving bijection between L_π^{geod} and the set of geodesics $\sigma \in S^*$ with $p_\sigma \in \textsc{Path}_\pi$. From Proposition 4.3 we see that $E^{-1}(\mathcal{G}_\pi)$ is a polyhedral set and thus each L_π^{geod} is a polyhedrally constrained language, as studied in Section 4.2. Then, from Proposition 4.5 we see that the multivariate generating function $f_\pi(x_1, x_2, \ldots, x_{|\Sigma|})$ of each L_π^{geod} is holonomic.

Let $a_{\pi,i} \in \mathbb{N}_+$ be the weight of the letter that corresponds to the variable x_i in the generating function $f_\pi(x_1, x_2, \ldots, x_{|\Sigma|})$. Let $h_\pi(z) \in \mathbb{C}[[z]]$ be defined as

$$h_\pi(z) = f_\pi(z^{a_{\pi,1}}, z^{a_{\pi,2}}, \ldots, z^{a_{\pi,|\Sigma|}}).$$

Then we see that the coefficient of z^k in $h_\pi(z)$ counts the geodesics $\sigma \in S^*$ for which $p_\sigma \in \textsc{Path}_\pi$ and $\omega(\sigma) = k$. Let $g(z) \in \mathbb{C}[[z]]$ be defined as

$$g(z) = \frac{1}{1-z} \cdot \sum_{\pi \in \textsc{Patt}} h_\pi(z),$$

Then we see that the coefficient of z^k in $g(z)$ is given by $\gamma_S(k)$, that is, $g(z)$ is the geodesic growth series $g(z) = \sum_{k=0}^{\infty} \gamma_S(k) z^k$. Moreover, from the closure properties in Lemma 2.9 we see that the function $g(z)$ is holonomic.

Our result then follows from Lemma 5.18. □

5.3. Language of Geodesics

In the previous section, we characterised the geodesic growth of virtually abelian groups. In our proof of this result, we found a bijection between the geodesics of the virtually abelian group and a finite union of formal languages. It is then natural to ask if there is a formal language characterisation for the language of geodesics. In this section we show that the language of geodesics can be recognised by blind multicounter automaton. Informally, we prove this result in Theorem C by implementing Algorithm 5.15 on a blind multicounter automaton, we then check if the word is geodesic using Lemma 5.17.

Theorem C. *The language of geodesics of a virtually abelian group with respect to any finite weighted monoid generating set S is blind multicounter.*

Proof. Let G be a virtually abelian group that is generated as a monoid by a finite weighted set S, and let $\mathbb{Z}^n \triangleleft G$ with finite index $d = [G : \mathbb{Z}^n]$. Let $\sigma \in S^*$, then from Algorithm 5.15 we have a patterned word $(v, \pi) = \text{Shuffle}(\sigma)$ for which $v^\pi \simeq \sigma$ and thus σ is a geodesic if and only if $v \in \mathcal{G}_\pi$ where $\mathcal{G}_\pi \subseteq \mathcal{N}_\pi$ is the polyhedral set given by Lemma 5.17.

The idea of our proof is to simulate Algorithm 5.15 on a blind multicounter automaton, while maintaining enough information on the machine's counters so that we may verify the membership of the vector v to the set $\mathcal{G}_\pi$.

For each polyhedral set $\mathcal{G}_\pi$, we fix a finite union of basic polyhedral sets

$$\mathcal{G}_\pi = \bigcup_{i=1}^{N_\pi} \mathcal{B}_{\pi,i}.$$

Then, for each basic polyhedral set $\mathcal{B}_{\pi,i}$, we fix a finite intersection

$$\begin{aligned}\mathcal{B}_{\pi,i} = \bigcap_{j=1}^{K_{\pi,i,1}} \{z \in \mathcal{Z}_\pi \,|\, \alpha_{\pi,i,j} \cdot z > \beta_{\pi,i,j}\} \\ \cap \bigcap_{j=1}^{K_{\pi,i,2}} \{z \in \mathcal{Z}_\pi \,|\, \chi_{\pi,i,j} \cdot z \equiv \eta_{\pi,i,j} \pmod{\theta_{\pi,i,j}}\} \\ \cap \bigcap_{j=1}^{K_{\pi,i,3}} \{z \in \mathcal{Z}_\pi \,|\, \xi_{\pi,i,j} \cdot z = \lambda_{\pi,i,j}\} \quad (5.12)\end{aligned}$$

where $\alpha_{\pi,i,j}, \chi_{\pi,i,j}, \xi_{\pi,i,j} \in \mathcal{Z}_\pi$, $\beta_{\pi,i,j}, \eta_{\pi,i,j}, \lambda_{\pi,i,j} \in \mathbb{Z}$ and $\theta_{\pi,i,j} \in \mathbb{N}_+$.

Let $k \in \mathbb{N}$ be such that $k \geqslant K_{\pi,i,1} + K_{\pi,i,2} + K_{\pi,i,3}$ for each basic polyhedral set $\mathcal{B}_{\pi,i}$. In the remainder of this proof, we construct a blind k-counter automaton $M = (Q, S, \delta, q_0, F, \mathfrak{e})$ that recognises the language of geodesics. Notice that the input alphabet of the machine is the generating set S.

For each basic polyhedral set $\mathcal{B}_{\pi,i}$, we define a map $C_{\pi,i}\colon \mathcal{N}_\pi \to \mathbb{Z}^k$ as

$$\begin{aligned} C_{\pi,i}(v) = (&\alpha_{\pi,i,1}\cdot v,\ \alpha_{\pi,i,2}\cdot v,\ \ldots,\ \alpha_{\pi,i,K_{\pi,i,1}}\cdot v,\\ &\chi_{\pi,i,1}\cdot v,\ \chi_{\pi,i,2}\cdot v,\ \ldots,\ \chi_{\pi,i,K_{\pi,i,2}}\cdot v,\\ &\xi_{\pi,i,1}\cdot v,\ \xi_{\pi,i,2}\cdot v,\ \ldots,\ \xi_{\pi,i,K_{\pi,i,3}}\cdot v, 0, 0, \ldots, 0). \end{aligned} \tag{5.13}$$

Notice that a vector $v \in \mathcal{N}_\pi$ belongs to $\mathcal{B}_{\pi,i}$ if and only if

$$C_{\pi,i}(v) = (a_1, a_2, \ldots, a_{K_{\pi,i,1}}, b_1, b_2, \ldots, b_{K_{\pi,i,2}}, c_1, c_2, \ldots, c_{K_{\pi,i,3}}, 0, 0, \ldots, 0)$$

where each $a_j > \beta_{\pi,i,j}$, each $b_j \equiv \eta_{\pi,i,j} \pmod{\theta_{\pi,i,j}}$ and each $c_j = \lambda_{\pi,i,j}$.

State-Space of the Machine.

For each $\tau \in \textsc{Patt}$, each basic polyhedral set $\mathcal{B}_{\pi,i}$, and each word $w \in S^*$ with $|w|_S \leqslant d$, we have a state of the form $[\tau, w, \pi, i] \in Q$. From these states, the machine will perform Algorithm 5.15 on its input word.

During the construction of our machine, we will ensure that if

$$(q_0, (0, 0, \ldots, 0), \sigma\mathfrak{e}) \vdash^* ([\tau, w, \pi, i], (c_1, c_2, \ldots, c_k), \zeta\mathfrak{e}),$$

then there is a $u \in \mathcal{N}_\tau$ with $u^\tau w\zeta \simeq \sigma$ and $(c_1, c_2, \ldots, c_k) = C_{\pi,i}(\mathrm{Proj}_{\tau,\pi}(u))$. In particular, this vector will correspond to some vector $u^{(i)}$ in the sequence of extended patterned words given in (5.6) as constructed in Algorithm 5.15.

For each basic polyhedral set $\mathcal{B}_{\pi,i}$, we have an accepting state $q_{\pi,i} \in F$. Moreover, our construction will have the property that $(q_0, \mathbf{0}, \sigma\mathfrak{e}) \vdash^* (q_{\pi,i}, \mathbf{0}, \mathfrak{e})$ if and only if $\mathrm{Shuffle}(\sigma) = (v, \pi)$ with $v \in \mathcal{B}_{\pi,i}$, and thus the machine M will accept the word σ if and only if it is a geodesic.

Nondeterministically Guessing a Basic Polyhedral Set.

The machine M begins simulating the word shuffling algorithm after nondeterministically guessing a basic polyhedral set $\mathcal{B}_{\pi,i}$ for which $\mathrm{Shuffle}(\sigma) = (v, \pi)$ with $v \in \mathcal{B}_{\pi,i}$. Notice that such a choice of basic polyhedral set exists if and only if σ is a geodesic. We accomplish this by introducing a relation

$$((q_0, \varepsilon), ([\varepsilon, \varepsilon, \pi, i], \mathbf{0})) \in \delta$$

for each basic polyhedral set $\mathcal{B}_{\pi,i}$, that is, we have a transition

$$(q_0, (0, 0, \ldots, 0), \sigma\mathfrak{e}) \vdash ([\varepsilon, \varepsilon, \pi, i], (0, 0, \ldots, 0), \sigma\mathfrak{e}) \tag{5.14}$$

for each $\mathcal{B}_{\pi,i}$. Notice that $\mathbf{0}^\varepsilon\sigma \simeq \sigma$ and $(0, 0, \ldots, 0) = C_{\pi,i}(\mathrm{Proj}_{\varepsilon,\pi}(\mathbf{0}))$.

Performing the Word Shuffling Algorithm.

For each extended strongly patterned word $((u^{(i)}, \tau^{(i)}), \sigma^{(i)})$ in sequence (5.6) in Algorithm 5.15, we will see that M has configurations of the form

$$([w, \tau^{(i)}, \pi, i], C_{\pi,i}(u^{(i)}), \zeta)$$

where $\sigma^{(i)} = w\zeta$. In order to apply the map Δ from such a configuration we will require that $w = \text{Prefix}(\sigma)$. We do so by introducing transitions as follows.

Let $([\tau, w, \pi, i], (c_1, c_2, \ldots, c_k), \zeta\mathfrak{e})$ be a configuration of M and let $\sigma = w\zeta$, then $w = \text{Prefix}(\sigma)$ if and only if either $|w|_S = d$ or $\zeta = \varepsilon$. Thus, for each word $w \in S^*$ with $|w|_S < d$ and each $s \in S$, we introduce a relation of the form

$$(([\tau, w, \pi, i], s), ([\tau, ws, \pi, i], \mathbf{0})) \in \delta$$

for each τ, π, i. From these relations we have a unique partial computation

$$([\tau, w, \pi, i], (c_1, c_2, \ldots, c_k), \zeta\mathfrak{e}) \vdash^* ([\tau, w', \pi, i], (c_1, c_2, \ldots, c_k), \zeta'\mathfrak{e}) \tag{5.15}$$

where $w' = \text{Prefix}(\sigma)$ and $\sigma = w\zeta = w'\zeta'$. We then apply the map Δ as follows.

Let $\tau \in \textsc{StrPatt}$ be a strong pattern, let $w, \zeta \in S^*$ with $w = \text{Prefix}(w\zeta)$ and $|w|_S \geqslant 1$, and let $\Delta(\tau, w) = (x, \tau', w')$. From Lemma 5.14, we see that for each vector $u \in \mathcal{N}_\tau$ we have $(u')^{\tau'} w'\zeta \simeq u^\tau w\zeta$ where $u' = \text{Proj}_{\tau,\tau'}(u) + e_{\tau',x}$. Moreover, we see that

$$C_{\pi,i}(\text{Proj}_{\tau,\pi}(u')) = C_{\pi,i}(\text{Proj}_{\tau,\pi}(u)) + C_{\pi,i}(\text{Proj}_{\tau',\pi}(e_{\tau',x})).$$

Notice that $w = \text{Prefix}(w\zeta)$ if and only if either $|w|_S = d$ or $\zeta = \varepsilon$. If $|w|_S = d$, then we introduce the relation

$$(([\tau, w, \pi, i], \varepsilon), ([\tau', w', \pi, i], C_{\pi,i}(\text{Proj}_{\tau',\pi}(e_{\tau',x})))) \in \delta$$

for each π, i; otherwise, if $|w|_S < d$, then we introduce the relation

$$(([\tau, w, \pi, i], \mathfrak{e}), ([\tau', w', \pi, i], C_{\pi,i}(\text{Proj}_{\tau',\pi}(e_{\tau',x})))) \in \delta$$

for each π, i. That is, we may apply the map Δ with the above relations.

Combining these transitions with those described in (5.15), we see that after nondeterministically choosing a basic polyhedral set in (5.14), the machine will deterministically perform Algorithm 5.15, then enter a configuration of the form

$$([\tau, \varepsilon, \pi, i], (c_1, c_2, \ldots, c_k), \mathfrak{e}) \tag{5.16}$$

with $(c_1, c_2, \ldots, c_k) = C_{\pi,i}(v)$ where $(v, \tau) = \text{Shuffle}(\sigma)$.

For each pair of patterns π, τ with $\pi \neq \tau$, and each basic polyhedral set $\mathcal{B}_{\pi,i}$, the machine has no transitions out of any configuration $([\tau, \varepsilon, \pi, i], c, \mathfrak{e})$ where $c \in \mathbb{Z}^k$. Hence, if the computation enters such a state, it cannot continue to an accepting configuration. Thus, we may assume without loss of generality that the machine nondeterministically chose the basic polyhedral set $\mathcal{B}_{\pi,i}$ with $\pi = \tau$ when performing the transition in (5.14). In the rest of our proof, we describe how the machine verifies that $v \in \mathcal{B}_{\pi,i}$.

Checking Polyhedral Set Membership.

Suppose that

$$(q_0, (0, 0, \ldots, 0), \sigma\mathfrak{e}) \vdash^* ([\pi, \varepsilon, \pi, i], (c_1, c_2, \ldots, c_k), \mathfrak{e}),$$

then $(c_1, c_2, \ldots, c_k) = C_{\pi,i}(v)$ where $(v, \pi) = \text{Shuffle}(\sigma)$. For each state of the form $[\pi, \varepsilon, \pi, i]$, we introduce a relation

$$(([\pi, \varepsilon, \pi, i], \mathfrak{e}), (q_{\pi,i}, \mu_{\pi,i})) \in \delta$$

where

$$\begin{aligned}\mu_{\pi,i} = (&-\beta_{\pi,i,1} - 1, -\beta_{\pi,i,2} - 1, \ldots, -\beta_{\pi,i,k} - 1,\\ &- \eta_{\pi,i,1}, -\eta_{\pi,i,2}, \ldots - \eta_{\pi,i,k},\\ &- \lambda_{\pi,i,1}, -\lambda_{\pi,i,2}, \ldots - \lambda_{\pi,i,k}, 0, 0, \ldots, 0).\end{aligned}$$

From this relation we have

$$([\pi, \varepsilon, \pi, i], (c_1, c_2, \ldots, c_k), \mathfrak{e}) \vdash (q_{\pi,i}, (c'_1, c'_2, \ldots, c'_k), \mathfrak{e})$$

where $v \in \mathcal{B}_{\pi,i}$ if and only if $(c'_1, c'_2, \ldots, c'_k)$ belongs to the set

$$\mathbb{N}^{K_{\pi,i,1}} \times \theta_{\pi,i,1}\mathbb{Z} \times \theta_{\pi,i,2}\mathbb{Z} \times \cdots \times \theta_{\pi,i,K_{\pi,i,2}}\mathbb{Z} \times \{0\}^{k-K_{\pi,i,1}-K_{\pi,i,2}}.$$

We verify v's membership to $\mathcal{B}_{\pi,i}$ by introducing additional relations as follows. For each $1 \leqslant j \leqslant K_{\pi,i,1}$, we have

$$((q_{\pi,i}, \mathfrak{e}), (q_{\pi,i}, -e_j)) \in \delta,$$

where $e_j \in \mathbb{Z}^k$ is the j-th standard basis element, and for each $1 \leqslant j \leqslant K_{\pi,i,2}$

$$((q_{\pi,i}, \mathfrak{e}), (q_{\pi,i}, \pm\theta_{\pi,i,j}\, e_{j'})) \in \delta$$

where $j' = K_{\pi,i,1} + j$ and $e_{j'} \in \mathbb{Z}^k$ is the j'-th standard basis element. From these relations, we see that we see that

$$(q_0, (0, 0, \ldots, 0), \sigma\mathfrak{e}) \vdash^* (q_{\pi,i}, (0, 0, \ldots, 0), \mathfrak{e}),$$

if and only if $v \in \mathcal{B}_{\pi,i}$ where $(v, \pi) = \text{Shuffle}(\sigma)$. □

5.4. Concluding Remarks

In this chapter we characterised the geodesic growth for all virtually abelian groups with respect to every generating set. Moreover, the proofs in this chapter are constructive, i.e., it is possible to compute the geodesic growth series for any given virtually abelian group.

Bridson, Burillo, Elder and Šunić gave a sufficient condition for a virtually abelian groups to have polynomial geodesic growth (see Lemma 5.1). It would be interesting to see if this condition is also necessary, that is, we ask the following question.

Question 5.23. *Let G be a virtually abelian group with polynomial geodesic growth. Then, is there an element $g \in G$ whose normal closure is a finite-index abelian subgroup of G?*

Recall that the geodesic growth is bounded from below by the volume growth, and thus, any group with polynomial geodesic growth must be virtually nilpotent. In this chapter, we have shown that for any abelian group A, there is a virtually-A group with polynomial geodesic growth (see the presentation in Equation (5.2)); and we have provided a method to determine if a virtually abelian group has polynomial geodesic growth. Until now, the only known examples of polynomial geodesic growth have been virtually abelian, e.g., the groups studied by Bridson, Burillo, Elder and Šunić [19]. It is then natural to ask if every group with polynomial geodesic growth is virtually abelian, and if not, then what counterexamples are there. In Chapter 6, we take the next step towards answering this question, and in moving forward to a classification of polynomial geodesic growth.

Chapter 6

Towards Virtually Nilpotent Groups

In Chapter 5 we characterised the geodesic growth of virtually abelian groups. In this chapter, we take the next step towards a classification of polynomial geodesic growth by furnishing an example of a virtually 2-step nilpotent group with polynomial geodesic growth. This is the first group which has been shown to have polynomial geodesic growth and is not virtually abelian. This result is important as it shows that a classification of polynomial geodesic growth must include groups beyond the class of virtually abelian.

In Theorem B, we show that there is a group that is virtually 2-step nilpotent and has polynomial geodesic growth. Our proof relies on a result that is implicit in the work of Blachère [16] which we provide in Lemma 6.1.

This result shows that there are groups with subexponential geodesic growth which are not virtually abelian, in particular, this example opens the door to the possible existence of a virtually nilpotent group that has intermediate geodesic growth with respect to some generating set. It also raises the question of whether polynomial geodesic growth is restricted to virtually nilpotent groups of step at most 2.

6.1. A Virtually Heisenberg Group

The integer Heisenberg group is the group of 3×3 upper-triangular integer matrices with 1's on their diagonals, that is, the group generated by the matrices

$$a = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

It is well known that the integer Heisenberg group, H_3, has the presentation

$$H_3 = \langle a, b \mid [a,[a,b]] = [b,[a,b]] = 1 \rangle .$$

Let X denote the standard generating set $X = \{a, a^{-1}, b, b^{-1}\}$ for H_3, following the convention of Blachère [16] we write $(x, y, z) \in H_3$ for the element corresponding to the

normal form word $[a,b]^z b^y a^x$. Inspired by the polynomial geodesic growth example in Equation (5.1), we construct a virtually Heisenberg group $H_3 \rtimes C_2$ as follows.

$$H_3 \rtimes C_2 = \langle a, b, t \mid [a,[a,b]] = [b,[a,b]] = t^2 = 1,\ a^t = b \rangle . \tag{6.1}$$

From the relation $b = a^t$ and applying a Tietze transform, we see that $S = \{a, a^{-1}, t\}$ is a generating set for $H_3 \rtimes C_2$. We provide a partial picture of the Cayley graph of $H_3 \rtimes C_2$ in Figure 6.1. Informally, one may think of this group as two copies of H_3 "glued" together with a "twist" by t edges.

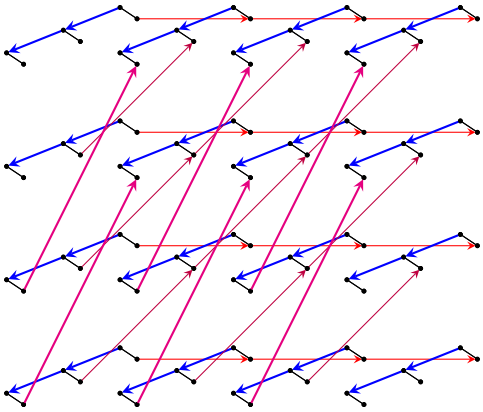

Figure 6.1: A Cayley graph for $H_3 \rtimes C_2$ with respect to the generating set S where the undirected edges are labelled by t and directed edges labelled by a.

Our goal is to show that any geodesic of $H_3 \rtimes C_2$ with respect to the generating set S can contain at most 7 instances of the letter t. From this we are able to place a polynomial upper bound on the geodesic growth function of $H_3 \rtimes C_2$. To do this, we first study geodesics of the integer Heisenberg group with respect to the generating set X.

Blachère [16, Theorem 2.2] provided explicit formulae for the length of elements in H_3, with respect the generating set X, by constructing geodesic representatives. We provide the following lemma which is implicit in the proof of Theorem 2.2 in [16].

Lemma 6.1. *Each element $(x,y,z) \in H_3$ has a geodesic representative with respect to the generating set $X = \{a, a^{-1}, b, b^{-1}\}$ of the form*

$$a^{\alpha_1} b^{\beta_1} a^{\alpha_2} b^{\beta_2} a^{\alpha_3} b^{\beta_3} \quad \text{or} \quad b^{\beta_1} a^{\alpha_1} b^{\beta_2} a^{\alpha_2} b^{\beta_3} a^{\alpha_3}$$

where each $\alpha_i, \beta_j \in \mathbb{Z}$.

Proof. We see that the lemma holds in the case of $(0,0,0) \in H_3$ as the empty word $\varepsilon \in S^*$ is such a geodesic. In the remainder of the proof, we assume that $(x,y,z) \neq (0,0,0)$. Following Blachère [16, p. 22] we reduce this proof to the case where $x, z \geqslant 0$ and $-x \leqslant y \leqslant x$ as follows.

Let $\tau \colon X^* \to X^*$ be the monoid isomorphism defined such that $\tau(a^k) = b^k$ and $\tau(b^k) = a^k$ for each $k \in \mathbb{Z}$. Let w^R denote the *reverse* of the word w, that is, if $w = w_1 w_2 \cdots w_k$ where each $w_j \in X$, then $w^R = w_k \cdots w_2 w_1$.

If $w \in X^*$ is a word as described in the lemma statement with $\overline{w} = (x, y, z)$, then $w' = \tau(w^R)$ is also in the form described in the lemma statement and $\overline{w'} = (y, x, z)$. Moreover, we see that $\tau(w^R)$ is a geodesic if and only if w is a geodesic. Defining the monoid isomorphisms $\varphi_a, \varphi_b \colon X^* \to X^*$ by $\varphi_a(a^k) = a^{-k}$, $\varphi_a(b^k) = b^k$, and $\varphi_b(a^k) = a^k$, $\varphi_b(b^k) = b^{-k}$ for each $k \in \mathbb{Z}$, we see that if $w \in X^*$ is a geodesic representative for $(x, y, z) \in H_3$, then $\varphi_a(w)$, $\varphi_b(w)$ and $\varphi_a(\varphi_b(w))$ are geodesics for $(-x, y, -z)$, $(x, -y, -z)$ and $(-x, -y, z)$, respectively, and each such word is in the form as described in the lemma statement. From application of the above transformations, we may assume without loss of generality that $x, z \geqslant 0$ and $-x \leqslant y \leqslant x$.

Let $h = (x, y, z) \in H_3$, then from [16, Theorem 2.2] we have the following formulae for the length $\ell_X(h)$ and (most importantly for us) geodesic representative for h.

I. If $y \geqslant 0$, then we have the following cases.

I.1. If $x < \sqrt{z}$, then $\ell_X(h) = 2\lfloor 2\sqrt{z} \rfloor - x - y$ and h has a geodesic representative given by $b^{y-y'} S_z a^{x-x'}$ where x', y' are the values given by $\overline{S_z} = (x', y', z)$ (cf. [16, p. 32]), where S_z is as follows.

- If $z = (n+1)^2$ for some $n \in \mathbb{N}$, then $S_z = a^{n+1} b^{n+1}$;
- if there exists a $k \in \mathbb{N}$ with $1 \leqslant k \leqslant n$ such that $z = n^2 + k$, then let $S_z = a^k b a^{n-k} b^n$;
- otherwise, there exists some $k \in \mathbb{N}$ with $1 \leqslant k \leqslant n$ such that $z = n^2 + n + k$ and we have $S_z = a^k b a^{n+1-k} b^n$.

I.2. If $x \geqslant \sqrt{z}$, then we have the following two cases:

I.2.1 $xy \geqslant z$, then $\ell_X(h) = x + y$, otherwise

I.2.2 $xy \leqslant z$, then $\ell_X(h) = 2\lceil z/x \rceil + x - y$;

and in both cases, the word $b^{y-u-1} a^v b a^{x-v} b^u$ is a geodesic for h where $0 \leqslant u$, $0 \leqslant v < x$ and $z = ux + v$ (cf. pages 24, 32 and 33 in [16]).

II. If $y < 0$, then we have the following cases.

II.1. If $x \leqslant \sqrt{z - xy}$, then $\ell_X(h) = 2\lceil 2\sqrt{z - xy} \rceil - x + y$. Let $n = \lceil \sqrt{z - xy} \rceil - 1$. Then

- there is either some $k \in \mathbb{N}$ with $1 \leqslant k \leqslant n$ such that we have $z - xy = n^2 + k$, and h has $a^{x-n} b^{-n-1} a^k b a^{n-k} b^{n+y}$ as a geodesic representative; or
- there is some $k \in \mathbb{N}$ with $0 \leqslant k \leqslant n$ such that we have $z - xy = (n+1)^2 - k$ and $a^{x-n} b^{-k} a^{-1} b^{k-n-1} a^{n+1} b^{n+1+y}$ is a geodesic representative for h (cf. [16, p. 24][1]).

II.2. If $x \geqslant \sqrt{z - xy}$, then $\ell_X(h) = 2\lceil z/x \rceil + x - y$ and h has a geodesic representative of $b^{y-u-1} a^v b a^{x-v} b^u$ where $u, v \geqslant 0$, $v < x$ and $z = ux + v$ (cf. [16, pp. 24 & 33]).

Notice that in each of the above cases, we have our desired result. □

From this lemma, we have the following result.

Corollary 6.1.1. *If $w \in S^*$ is a geodesic of $H_3 \rtimes C_2$ with respect to the generating set $S = \{a, a^{-1}, t\}$, then w contains at most 7 instances of the letter t.*

Proof. Let $w \in S^*$ be a word containing 8 instances of t of the form

$$w = a^{n_1} t a^{m_1} t a^{n_2} t a^{m_2} t a^{n_3} t a^{m_3} t a^{n_4} t a^{m_4} t,$$

[1] Note that in [16] there is an error in the second case.

where $n_i, m_i \in \mathbb{Z}$, and notice that $\overline{w}$ belongs to the subgroup H_3. The Tietze transform given by $b = tat$ which we applied to obtain the generating set $S = \{a, a^{-1}, t\}$ from (6.1) yields an automorphism $\varphi\colon H_3 \rtimes C_2 \to H_3 \rtimes C_2$ given by $\varphi(a) = a$, $\varphi(t) = t$, $\varphi(b) = tat$, and since $t^2 = 1$ we have $\varphi(b^k) = ta^k t$ for $k \in \mathbb{Z}$. Let $X = \{a, a^{-1}, b, b^{-1}\}$ be a generating set for the subgroup H_3. Then from the word $w \in S^*$ we may construct a word

$$w_2 = a^{n_1} b^{m_1} a^{n_2} b^{m_2} a^{n_3} b^{m_3} a^{n_4} b^{m_4} \in X^*$$

where $\overline{w_2} = \overline{w}$ since $\varphi(w_2) = w$. Moreover, $|w|_S = |w_2|_X + 8$.

From Lemma 6.1, we know that there is a word $w_3 \in X^*$, with $\overline{w_3} = \overline{w_2}$ and $|w_3|_X \leqslant |w_2|_X$, of the form

$$w_3 = a^{\alpha_1} b^{\beta_1} a^{\alpha_2} b^{\beta_2} a^{\alpha_3} b^{\beta_3} \quad \text{or} \quad w_3 = b^{\beta_1} a^{\alpha_1} b^{\beta_2} a^{\alpha_2} b^{\beta_3} a^{\alpha_3}$$

where $\alpha_i, \beta_i \in \mathbb{Z}$ (possibly zero). We then see that $\overline{w}$ can be represented by a word of the form

$$w_4 = a^{\alpha_1} t a^{\beta_1} t a^{\alpha_2} t a^{\beta_2} t a^{\alpha_3} t a^{\beta_3} t \quad \text{or} \quad w_4 = t a^{\beta_1} t a^{\alpha_1} t a^{\beta_2} t a^{\alpha_2} t a^{\beta_3} t a^{\alpha_3}$$

where

$$|w_4|_S = |w_3|_X + 6 < |w_2|_X + 8 = |w|_S.$$

Then w cannot be a geodesic as we have a strictly shorter word w_4 that represents the same element. Thus, a geodesic of $H_3 \rtimes C_2$ with respect to $S = \{a, a^{-1}, t\}$ can contain at most 7 instances of the letter t as we can replace any subword with 8 instances of t with a strictly shorter word containing at most 7 instances of t. □

From this corollary we may immediately obtain the following polynomial upper bound on the geodesic growth function.

Theorem B. *The geodesic growth function of $H_3 \rtimes C_2$ with respect to $S = \{a, a^{-1}, t\}$ is bounded from above by a polynomial of degree 8.*

Proof. From Corollary 6.1.1, we see that any geodesic of $H_3 \rtimes C_2$, with respect to the generating set S, must have the form

$$w = a^{m_1} t a^{m_2} t \cdots t a^{m_{k+1}}$$

where $k \leqslant 7$ and each $m_i \in \mathbb{Z}$. Then with k fixed and $r = |w|_S$, we see that there are at most 2^{k+1} choices for the sign of $m_1, m_2, \ldots, m_{k+1}$, and at most $\binom{r}{k}$ choices for the placement of the t's in w. Thus, the geodesic growth function $\gamma_S(n)$ has an upper bound given by

$$\gamma_S(n) \leqslant \sum_{k=0}^{7} \sum_{r=k}^{n} 2^{k+1} \binom{r}{k} \leqslant \sum_{k=0}^{7} \sum_{r=k}^{n} 2^{k+1} r^k \leqslant 8 \cdot 2^8 n^8$$

which gives the degree 8 polynomial upper bound. □

6.2. Concluding Remarks and Open Questions

The proof that the virtually 2-step nilpotent group in this chapter has polynomial geodesic growth relied heavily on work of Blachère (see Lemma 6.1) and does not appear to be generalisable in its current form. It is then natural to ask if there are nilpotent groups of higher step with analogous properties, in particular, we ask the following question.

Question 6.2. *Is there a virtually k-step nilpotent group with polynomial geodesic growth for some $k \geqslant 3$, and if so, is there such an example for each k?*

It follows from the work of Bass [9, Theorem 2] that the usual growth rate of a virtually nilpotent group is polynomial of integer degree. Moreover, from Theorem A it is known that if a virtually abelian group has polynomial geodesic growth, then it must be of integer degree since the geodesic growth series is rational in this case. It is not known if there is a group with polynomial geodesic growth of a non-integer degree.

Question 6.3. *Is there a group with polynomial geodesic growth of a non-integer degree?*

Based on experimental results (see [12]) we conjecture that the geodesic growth rate of $H_3 \rtimes C_2$ with respect to the generating set S can be bounded from above and below by polynomials of degree six (cf. the volume growth is polynomial of degree four). We ask the following question.

From Theorem A we know that if a virtually abelian group has polynomial geodesic growth, then its geodesic growth series is rational. However, it is unclear if this property is held by virtually nilpotent groups. From experimental results, it appears that the geodesic growth series of $H_3 \rtimes C_2$ with respect to S is not rational (see [12]).

Question 6.4. *Is the geodesic growth series for $H_3 \rtimes C_2$ with respect to S rational?*

In this thesis we have taken steps towards a classification of polynomial geodesic growth, and more generally towards the study of the geodesic growth of virtually nilpotent groups. In particular, we characterised the geodesic growth of virtually abelian groups; and provided the first example of a group with polynomial geodesic growth that is not virtually abelian. The results in this thesis open up new questions and new techniques for obtaining characterisations.

Appendix A

Additional Proofs

In this appendix we provide proofs of incidental statements made throughout this thesis. These proofs are provided in the interest of completeness.

In the introduction, it was stated that the word problem for any finitely presented group is recursively enumerable. This is a well-known fact which we prove as follows.

Proposition A.1. *Finitely presented groups have recursively enumerable word problems.*

Proof. Let G be a group with presentation $\langle X \mid R \rangle$ where X and R are both finite. We then see that a word $w \in X^*$ is in the word problem if and only if we have

$$w =_{F_X} \prod_{i=0}^{n} u_i r_i^{\delta_i} u_i^{-1}$$

for some $n \in \mathbb{N}$ where each $u_i \in X^*$, $\delta_j \in \{-1, 1\}$ and $r_i \in R$. Notice here that '$=_{F_X}$' denotes that the left and right-hand sides are the same word after free-reduction is performed, that is, they are equivalent if viewed as elements of the free group.

We may then construct a Turing machine M which takes a word $w \in X^*$ as input, then iterates through the set of all finite products $\prod_{i=0}^{n} u_i r_i^{\delta_i} u_i^{-1}$. At each iteration, the machine should compare the word w and the result of the product. The machine then terminate and accepts only if the two words are equal.

We see that the machine M accepts a word $w \in X^*$ if and only if it lies within the word problem WP_X, that is, membership to the word problem WP_X is *semi-decidable*.

It is well known that a problem is semi-decidable if and only if the set of all accepted words (in this case the word problem) is recursively enumerable. This can be proven by constructing a machine which checks all words in parallel using a technique known as *dovetailing* (see Theorem 20.8 on p. 441 of [85] for a proof of this fact). □

In Section 2.3.1 we gave an example of a linearly constrained language, and provided its generating function. In the following we show that this generating function is holonomic by explicitly constructing a system of linear differential equations which it satisfies.

Proposition A.2. *The multivariate power series*

$$f(x,y,z) = \sum_{n\in\mathbb{N}} \frac{(3n)!}{(n!)^3} x^n y^n z^n$$

satisfies the differential equations

$$\left.\begin{aligned}
(x^2 - 27x^3yz)\partial_x^2 f(x,y,z) + (x - 54x^2yz)\partial_x f(x,y,z) - 6xyz\, f(x,y,z) = 0\\
(y^2 - 27xy^3z)\partial_y^2 f(x,y,z) + (y - 54xy^2z)\partial_y f(x,y,z) - 6xyz\, f(x,y,z) = 0\\
(z^2 - 27xyz^3)\partial_z^2 f(x,y,z) + (z - 54xyz^2)\partial_z f(x,y,z) - 6xyz\, f(x,y,z) = 0.
\end{aligned}\right\} \tag{A.1}$$

Thus, $f(x,y,z)$ is holonomic.

Proof. Notice that the system of differential equations in (A.1) is equivalent to

$$\left.\begin{aligned}
x^2\partial_x^2 f(x,y,z) + x\partial_x f(x,y,z) - 27x^2\partial_x^2(xyz\, f(x,y,z)) - 6xyz\, f(x,y,z) = 0\\
y^2\partial_y^2 f(x,y,z) + y\partial_y f(x,y,z) - 27y^2\partial_y^2(xyz\, f(x,y,z)) - 6xyz\, f(x,y,z) = 0\\
z^2\partial_z^2 f(x,y,z) + z\partial_z f(x,y,z) - 27z^2\partial_z^2(xyz\, f(x,y,z)) - 6xyz\, f(x,y,z) = 0.
\end{aligned}\right\} \tag{A.2}$$

This can be shown using the *product rule* of differentiation.

Let

$$f(x,y,z) = \sum_{n\in\mathbb{N}} \frac{(3n)!}{(n!)^3} x^n y^n z^n,$$

then

$$\begin{aligned}
x\partial_x f(x,y,z) &= \sum_{n=1}^{\infty} \frac{(3n)!}{(n!)^3}\, n\, x^n y^n z^n,\\
x^2\partial_x^2 f(x,y,z) &= \sum_{n=1}^{\infty} \frac{(3n)!}{(n!)^3}\, n(n-1)\, x^n y^n z^n,\\
xyz\, f(x,y,z) &= \sum_{n=1}^{\infty} \frac{(3(n-1))!}{((n-1)!)^3}\, x^n y^n z^n, \text{ and}\\
x^2\partial_x^2(xyz\, f(x,y,z)) &= \sum_{n=1}^{\infty} \frac{(3(n-1))!}{((n-1)!)^3}\, n(n-1)\, x^n y^n z^n.
\end{aligned}$$

We then see that

$$x^2\partial_x^2 f(x,y,z) + x\partial_x f(x,y,z) - 27x^2\partial_x^2(xyz\, f(x,y,z)) - 6xyz\, f(x,y,z) = \sum_{n=1}^{\infty} c_n x^n y^n z^n$$

where each

$$\begin{aligned}
c_n &= n^2\frac{(3n)!}{(n!)^3} - (27n^2 - 27n + 6)\frac{(3(n-1))!}{((n-1)!)^3}\\
&= \frac{1}{n}\left[n^3\frac{(3n)!}{(n!)^3} - 3n(3n-1)(3n-2)\frac{(3(n-1))!}{((n-1)!)^3}\right].
\end{aligned}$$

Moreover, we see that each $c_n = 0$ as

$$\frac{(3n)!}{(n!)^3} = \frac{3n(3n-1)(3n-2)}{n^3} \cdot \frac{(3(n-1))!}{((n-1)!)^3}.$$

Thus, we have

$$x^2\partial_x^2 f(x,y,z) + x\partial_x f(x,y,z) - 27x^2\partial_x^2(xyz\, f(x,y,z)) - 6xyz\, f(x,y,z) = 0.$$

The proofs of the other two differential equations in (A.2) are the same. □

References

[1] Alfred V. Aho. 'Indexed grammars—an extension of context-free grammars'. In: *J. Assoc. Comput. Mach.* 15 (1968), pp. 647–671. ISSN: 0004-5411. DOI: 10.1145/321479.321488.

[2] A. V. Anisimov. 'The group languages'. In: *Kibernetika (Kiev)* 7.4 (1971), pp. 18–24. ISSN: 0023-1274.

[3] Yago Antolín and Laura Ciobanu. 'Geodesic growth in right-angled and even Coxeter groups'. In: *European J. Combin.* 34.5 (July 2013), pp. 859–874. ISSN: 0195-6698. DOI: 10.1016/j.ejc.2012.12.007.

[4] Yago Antolín and Islam Foniqi. *Geodesic Growth of some 3-dimensional RACGs*. 2021. arXiv: 2105.09751 [math.GR].

[5] Peter R. J. Asveld. 'Controlled iteration grammars and full hyper-AFL's'. In: *Information and Control* 34.3 (1977), pp. 248–269. ISSN: 0890-5401.

[6] Jayadev S. Athreya and Amritanshu Prasad. *Growth in Right-Angled Groups and Monoids*. 2014. arXiv: 1409.4142 [math.GR].

[7] Louis Auslander. 'On a problem of Philip Hall'. In: *Ann. of Math. (2)* 86 (1967), pp. 112–116. ISSN: 0003-486X. DOI: 10.2307/1970362.

[8] Brenda S. Baker and Ronald V. Book. 'Reversal-bounded multipushdown machines'. In: *J. Comput. System Sci.* 8 (1974), pp. 315–332. ISSN: 0022-0000. DOI: 10.1016/S0022-0000(74)80027-9.

[9] H. Bass. 'The degree of polynomial growth of finitely generated nilpotent groups'. In: *Proc. London Math. Soc. (3)* 25 (1972), pp. 603–614. ISSN: 0024-6115. DOI: 10.1112/plms/s3-25.4.603.

[10] M. Benson. 'Growth series of finite extensions of $\mathbb{Z}^n$ are rational'. In: *Invent. Math.* 73.2 (1983), pp. 251–269. ISSN: 0020-9910. DOI: 10.1007/BF01394026.

[11] Rose Berns-Zieve, Dana Fry, Johnny Gillings, Hannah Hoganson and Heather Mathews. 'Groups with context-free co-word problem and embeddings into Thompson's group V'. In: London Math. Soc. Lecture Note Ser. 451 (2018), pp. 19–37.

[12] Alex Bishop. *A virtually 2-step nilpotent group with polynomial geodesic growth (data and code)*. July 2020. DOI: 10.5281/zenodo.3941381.

[13] Alex Bishop. 'Geodesic growth in virtually abelian groups'. In: *J. Algebra* 573 (2021), pp. 760–786. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2020.12.003.

[14] Alex Bishop and Murray Elder. *A virtually 2-step nilpotent group with polynomial geodesic growth*. 2020. arXiv: 2007.06834 [math.GR].

[15] Alex Bishop and Murray Elder. 'Bounded automata groups are co-ET0L'. In: *Language and automata theory and applications*. Vol. 11417. Lecture Notes in Comput. Sci. Springer, Cham, 2019, pp. 82–94. DOI: 10.1007/978-3-030-13435-8_6.

[16] Sébastien Blachère. 'Word distance on the discrete Heisenberg group'. In: *Colloq. Math.* 95.1 (2003), pp. 21–36. ISSN: 0010-1354. DOI: 10.4064/cm95-1-2.

[17] Collin Bleak, Francesco Matucci and Max Neunhöffer. 'Embeddings into Thompson's group V and $co\mathcal{CF}$ groups'. In: *J. Lond. Math. Soc. (2)* 94.2 (2016), pp. 583–597. ISSN: 0024-6107. DOI: 10.1112/jlms/jdw044.

[18] William W. Boone. 'The word problem'. In: *Ann. of Math. (2)* 70 (1959), pp. 207–265. ISSN: 0003-486X. DOI: 10.2307/1970103.

[19] Martin R. Bridson, José Burillo, Murray Elder and Zoran Šunić. 'On groups whose geodesic growth is polynomial'. In: *Internat. J. Algebra Comput.* 22.5 (2012), pp. 1250048, 13. ISSN: 0218-1967. DOI: 10.1142/S0218196712500488.

[20] J. L. Britton. 'P. S. Novikov. Ob algoritmičéskoj nérazréšimosti problémy toždéstva slov v téorii grupp (Algorithmic unsolvability of the word problem in group theory). Trudy Matématičéskogo Instituta iméni V. A. Stéklova, vol. 44. Izdatél'stvo Akadémii Nauk SSSR, Moscow1955, 143 pp.' In: *Journal of Symbolic Logic* 23.1 (1958), pp. 50–52. DOI: 10.2307/2964487.

[21] Julie Marie Brönnimann. 'Geodesic growth of groups'. PhD thesis. Université de Neuchâtel, 2016. URL: http://doc.rero.ch/record/277391/files/00002547.pdf.

[22] Fritz Carlson. 'Über Potenzreihen mit ganzzahligen Koeffizienten'. In: *Math. Z.* 9.1-2 (1921), pp. 1–13. ISSN: 0025-5874. DOI: 10.1007/BF01378331.

[23] Giusi Castiglione and Paolo Massazza. 'On a class of languages with holonomic generating functions'. In: *Theoret. Comput. Sci.* 658.part A (2017), pp. 74–84. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2016.07.022.

[24] Ruth Charney and John Meier. 'The language of geodesics for Garside groups'. In: *Math. Z.* 248.3 (2004), pp. 495–509. ISSN: 0025-5874. DOI: 10.1007/s00209-004-0666-8.

[25] A. Noam Chomsky. *Context-free grammars and pushdown storage*. Quarterly Progress Report no. 65. Massachusetts Institute of Technology, Research Laboratory of Electronics, 15 Apr. 1962, pp. 187–194. URL: http://hdl.handle.net/1721.1/53697.

[26] N. Chomsky and M. P. Schützenberger. 'The algebraic theory of context-free languages'. In: *Computer programming and formal systems*. North-Holland, Amsterdam, 1963, pp. 118–161.

[27] Noam Chomsky. 'On certain formal properties of grammars'. In: *Information and Control* 2 (1959), pp. 137–167. ISSN: 0019-9958.

[28] P. A. Christensen. 'Hyper-AFL's and ET0L systems'. In: (1974), 254–257, 327–338. Lecture Notes in Comput. Sci., Vol. 15.

[29] Laura Ciobanu, Volker Diekert and Murray Elder. 'Solution sets for equations over free groups are EDT0L languages'. In: *Internat. J. Algebra Comput.* 26.5 (2016), pp. 843–886. ISSN: 0218-1967. DOI: 10.1142/S0218196716500363.

[30] Laura Ciobanu and Murray Elder. 'Solutions sets to systems of equations in hyperbolic groups are EDT0L in PSPACE'. In: *46th International Colloquium on Automata, Languages, and Programming*. Vol. 132. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2019, Art. No. 110, 15.

[31] Laura Ciobanu and Murray Elder. 'The complexity of solution sets to equations in hyperbolic groups'. In: *Israel Journal of Mathematics* (Nov. 2021). DOI: 10.1007/s11856-021-2232-z. URL: https://doi.org/10.1007/s11856-021-2232-z.

[32] Laura Ciobanu, Murray Elder and Michal Ferov. 'Applications of L systems to group theory'. In: *Internat. J. Algebra Comput.* 28.2 (2018), pp. 309–329. ISSN: 0218-1967. DOI: 10.1142/S0218196718500145.

[33] Laura Ciobanu and Alexander Kolpakov. 'Geodesic growth of right-angled Coxeter groups based on trees'. In: *J. Algebraic Combin.* 44.2 (2016), pp. 249–264. ISSN: 0925-9899. DOI: 10.1007/s10801-016-0667-9.

[34] Sean Cleary, Murray Elder and Jennifer Taback. 'Cone types and geodesic languages for lamplighter groups and Thompson's group F'. In: *J. Algebra* 303.2 (2006), pp. 476–500. ISSN: 0021-8693. DOI: 10.1016/j.jalgebra.2005.11.016.

[35] Donald J. Collins. 'A simple presentation of a group with unsolvable word problem'. In: *Illinois J. Math.* 30.2 (1986), pp. 230–234. ISSN: 0019-2082. URL: http://projecteuclid.org/euclid.ijm/1256044631.

[36] Karel Čulik. 'On some families of languages related to developmental systems'. In: *Internat. J. Comput. Math.* 4 (1974), pp. 31–42. ISSN: 0020-7160. DOI: 10.1080/00207167408803079.

[37] Volker Diekert and Murray Elder. 'Solutions of twisted word equations, EDT0L languages, and context-free groups'. In: *44th International Colloquium on Automata, Languages, and Programming*. Vol. 80. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2017, Art. No. 96, 14.

[38] Volker Diekert and Anca Muscholl. 'Solvability of equations in free partially commutative groups is decidable'. In: *Automata, languages and programming*. Vol. 2076. Lecture Notes in Comput. Sci. Springer, Berlin, 2001, pp. 543–554. DOI: 10.1007/3-540-48224-5_45.

[39] Moon Duchin and Michael Shapiro. 'The Heisenberg group is pan-rational'. In: *Adv. Math.* 346 (2019), pp. 219–263. ISSN: 0001-8708. DOI: 10.1016/j.aim.2019.01.046.

[40] A. Ehrenfeucht, G. Rozenberg and S. Skyum. 'A relationship between ET0L and EDT0L languages'. In: *Theoretical Computer Science* 1.4 (Apr. 1976), pp. 325–330. DOI: 10.1016/0304-3975(76)90076-1.

[41] M. Elder, M. Gutierrez and Z. Šunić. 'Geodesics in the first Grigorchuk group'. unpublished. 2006.

[42] Murray Elder, Mark Kambites and Gretchen Ostheimer. 'On groups and counter automata'. In: *Internat. J. Algebra Comput.* 18.8 (2008), pp. 1345–1364. ISSN: 0218-1967. DOI: 10.1142/S0218196708004901.

[43] David B. A. Epstein, James W. Cannon, Derek F. Holt, Silvio V. F. Levy, Michael S. Paterson and William P. Thurston. *Word processing in groups*. Jones and Bartlett Publishers, Boston, MA, 1992, pp. xii+330. ISBN: 0-86720-244-0.

[44] Alex Evetts. 'Rational growth in virtually abelian groups'. In: *Illinois J. Math.* 63.4 (2019), pp. 513–549. ISSN: 0019-2082. DOI: 10.1215/00192082-8011497.

[45] Alex Evetts and Alex Levine. *Equations in virtually abelian groups: languages and growth*. 2020. arXiv: 2009.03968 [math.GR].

[46] Robert James Evey. 'The theory and applications of pushdown store machines'. PhD thesis. Harvard University, May 1963.

[47] M. Fekete. 'Über die Verteilung der Wurzeln bei gewissen algebraischen Gleichungen mit ganzzahligen Koeffizienten'. German. In: *Math. Z.* 17.1 (1923), pp. 228–249. ISSN: 0025-5874. DOI: 10.1007/BF01504345.

[48] Philippe Flajolet. 'Analytic models and ambiguity of context-free languages'. In: vol. 49. 2-3. Twelfth international colloquium on automata, languages and programming (Nafplion, 1985). 1987, pp. 283–309. DOI: 10.1016/0304-3975(87)90011-9.

[49] Philippe Flajolet, Stefan Gerhold and Bruno Salvy. 'On the non-holonomic character of logarithms, powers, and the nth prime function'. In: *Electron. J. Combin.* 11.2 (Apr. 2005), Article 2, 16. URL: http://www.combinatorics.org/Volume_11/Abstracts/v11i2a2.html.

[50] Philippe Flajolet and Robert Sedgewick. *Analytic combinatorics*. Cambridge University Press, Cambridge, 2009, pp. xiv+810. ISBN: 978-0-521-89806-5. DOI: 10.1017/CBO9780511801655.

[51] S. A. Greibach. 'Formal languages: origins and directions'. In: *Ann. Hist. Comput.* 3.1 (1981), pp. 14–41. ISSN: 0164-1239. DOI: 10.1109/MAHC.1981.10006.

[52] S. A. Greibach. 'Remarks on blind and partially blind one-way multicounter machines'. In: *Theoret. Comput. Sci.* 7.3 (1978), pp. 311–324. ISSN: 0304-3975. DOI: 10.1016/0304-3975(78)90020-8.

[53] R. I. Grigorchuk. 'On the Milnor problem of group growth'. In: *Dokl. Akad. Nauk SSSR* 271.1 (1983), pp. 30–33. ISSN: 0002-3264.

[54] Rostislav Grigorchuk. 'Milnor's problem on the growth of groups and its consequences'. In: *Frontiers in complex dynamics*. Vol. 51. Princeton Math. Ser. Princeton Univ. Press, Princeton, NJ, 2014, pp. 705–773.

[55] Rostislav Grigorchuk. 'On Burnside's problem on periodic groups'. In: *Funktsional. Anal. i Prilozhen.* 14.1 (1980), pp. 53–54. ISSN: 0374-1990.

[56] Mikhael Gromov. 'Groups of polynomial growth and expanding maps'. In: *Inst. Hautes Études Sci. Publ. Math.* 53.53 (1981), pp. 53–73. ISSN: 0073-8301. URL: http://www.numdam.org/item?id=PMIHES_1981__53__53_0.

[57] Narain Gupta and Saïd Sidki. 'On the Burnside problem for periodic groups'. In: *Mathematische Zeitschrift* 182.3 (Sept. 1983), pp. 385–388. ISSN: 0025-5874. DOI: 10.1007/BF01179757.

[58] Michael A. Harrison. *Introduction to formal language theory*. Addison-Wesley Publishing Co., Reading, Mass., 1978, pp. xiv+594. ISBN: 0-201-02955-3.

[59] Susan Hermiller, Derek F. Holt and Sarah Rees. 'Groups whose geodesics are locally testable'. In: *Internat. J. Algebra Comput.* 18.5 (2008), pp. 911–923. ISSN: 0218-1967. DOI: 10.1142/S0218196708004676.

[60] Derek F. Holt and Sarah Rees. 'Artin groups of large type are shortlex automatic with regular geodesics'. In: *Proc. Lond. Math. Soc. (3)* 104.3 (2012), pp. 486–512. ISSN: 0024-6115. DOI: 10.1112/plms/pdr035.

[61] Derek F. Holt, Sarah Rees, Claas E. Röver and Richard M. Thomas. 'Groups with context-free co-word problem'. In: *J. London Math. Soc. (2)* 71.3 (2005), pp. 643–657. ISSN: 0024-6107. DOI: 10.1112/S002461070500654X.

[62] Derek F. Holt, Sarah Rees and Michael Shapiro. 'Groups that do and do not have growing context-sensitive word problem'. In: *International Journal of Algebra and Computation* 18.07 (Nov. 2008), pp. 1179–1191. DOI: 10.1142/S0218196708004834.

[63] Derek F. Holt and Claas E. Röver. 'Groups with indexed co-word problem'. In: *Internat. J. Algebra Comput.* 16.5 (2006), pp. 985–1014. ISSN: 0218-1967. DOI: 10.1142/S0218196706003359.

[64] Felix Klaedtke and Harald Rueß. 'Monadic second-order logics with cardinalities'. In: *Automata, languages and programming*. Vol. 2719. Lecture Notes in Comput. Sci. Springer, Berlin, 2003, pp. 681–696. DOI: 10.1007/3-540-45061-0_54.

[65] Felix Klaedtke and Harald Rueß. *Parikh Automata and Monadic Second-Order Logics with Linear Cardinality Constraints*. Technical Report 177. Albert-Ludwigs-Universität Freiburg, 2002. URL: http://tr.informatik.uni-freiburg.de/2002/Report177/index.php.

[66] S. C. Kleene. 'Representation of events in nerve nets and finite automata'. In: *Automata studies*. Annals of mathematics studies, no. 34. Princeton University Press, Princeton, N. J., 1956, pp. 3–41.

[67] Alexander Kolpakov and Alexey Talambutsa. 'Spherical and geodesic growth rates of right-angled Coxeter and Artin groups are Perron numbers'. In: *Discrete Math.* 343.3 (2020), pp. 111763, 8. ISSN: 0012-365X. DOI: 10.1016/j.disc.2019.111763.

[68] S.-Y. Kuroda. 'Classes of languages and linear-bounded automata'. In: *Information and Control* 7 (1964), pp. 207–223. ISSN: 0019-9958.

[69] Jan van Leeuwen. 'Variations of a new machine model'. In: *17th Annual Symposium on Foundations of Computer Science (Houston, Tex., 1976)*. IEEE Comput. Soc., Long Beach, Calif., Oct. 1976, pp. 228–235. DOI: 10.1109/SFCS.1976.35.

[70] J. Lehnert and P. Schweitzer. 'The co-word problem for the Higman-Thompson group is context-free'. In: *Bull. Lond. Math. Soc.* 39.2 (2007), pp. 235–241. ISSN: 0024-6093. DOI: 10.1112/blms/bdl043.

[71] Jörg Lehnert. 'Gruppen von quasi-Automorphismen'. In: *Goethe-Universitaet Frankfurt am Main* (2008).

[72] Aristid Lindenmayer. 'Mathematical models for cellular interactions in development I. Filaments with one-sided inputs'. In: *Journal of Theoretical Biology* 18.3 (Apr. 1968), pp. 280–99. DOI: 10.1016/0022-5193(68)90079-9.

[73] L. Lipshitz. 'D-finite power series'. In: *J. Algebra* 122.2 (1989), pp. 353–373. ISSN: 0021-8693. DOI: 10.1016/0021-8693(89)90222-6.

[74] Joseph Loeffler, John Meier and James Worthington. 'Graph products and Cannon pairs'. In: *Internat. J. Algebra Comput.* 12.6 (2002), pp. 747–754. ISSN: 0218-1967. DOI: 10.1142/S021819670200122X.

[75] Jean Mairesse and Frédéric Mathéus. 'Growth series for Artin groups of dihedral type'. In: *Internat. J. Algebra Comput.* 16.6 (2006), pp. 1087–1107. ISSN: 0218-1967. DOI: 10.1142/S0218196706003360.

[76] P. Massazza. 'Holonomic functions and their relation to linearly constrained languages'. In: *RAIRO Inform. Théor. Appl.* 27.2 (1993), pp. 149–161. ISSN: 0988-3754. DOI: 10.1051/ita/1993270201491.

[77] John Milnor. 'Advanced Problems: 5603'. In: *The American Mathematical Monthly* 75.6 (1968), pp. 685–686. ISSN: 00029890, 19300972. URL: http://www.jstor.org/stable/2313822.

[78] David E. Muller and Paul E. Schupp. 'Groups, the theory of ends, and context-free languages'. In: *J. Comput. System Sci.* 26.3 (1983), pp. 295–310. ISSN: 0022-0000. DOI: 10.1016/0022-0000(83)90003-X.

[79] Volodymyr Nekrashevych. *Self-similar groups*. Vol. 117. Mathematical Surveys and Monographs. American Mathematical Society, Providence, RI, 2005, pp. xii+231. ISBN: 0-8218-3831-8. DOI: 10.1090/surv/117.

[80] Walter D. Neumann and Michael Shapiro. 'Automatic structures, rational growth, and geometrically finite hyperbolic groups'. In: *Invent. Math.* 120.2 (1995), pp. 259–287. ISSN: 0020-9910. DOI: 10.1007/BF01241129.

[81] P. S. Novikov. *Ob algoritmičeskoĭ nerazrešimosti problemy toždestva slov v teorii grupp*. Trudy Mat. Inst. im. Steklov. no. 44. Izdat. Akad. Nauk SSSR, Moscow, 1955, p. 143.

[82] Emil L. Post. 'Formal reductions of the general combinatorial decision problem'. In: *Amer. J. Math.* 65 (1943), pp. 197–215. ISSN: 0002-9327. DOI: 10.2307/2371809.

[83] Emil L. Post. 'Recursive unsolvability of a problem of Thue'. In: *J. Symbolic Logic* 12 (1947), pp. 1–11. ISSN: 0022-4812. DOI: 10.2307/2267170.

[84] M. O. Rabin and D. Scott. 'Finite automata and their decision problems'. In: *IBM J. Res. Develop.* 3 (1959), pp. 114–125. ISSN: 0018-8646. DOI: 10.1147/rd.32.0114.

[85] Elaine A. Rich. *Automata, Computability and Complexity. Theory and Applications*. Pearson Prentice Hall, 2007. ISBN: 9780132288064.

[86] Derek J. S. Robinson. *A course in the theory of groups*. Second. Vol. 80. Graduate Texts in Mathematics. Springer-Verlag, New York, 1996, pp. xviii+499. ISBN: 0-387-94461-3. DOI: 10.1007/978-1-4419-8594-1.

[87] Grzegorz Rozenberg. 'Extension of tabled OL-systems and languages'. In: *Internat. J. Comput. Information Sci.* 2 (1973), pp. 311–336. ISSN: 0091-7036.

[88] Lucas Sabalka. 'Geodesics in the braid group on three strands'. In: *Group theory, statistics, and cryptography*. Vol. 360. Contemp. Math. Amer. Math. Soc., Providence, RI, 2004, pp. 133–150. DOI: 10.1090/conm/360/06575.

[89] Michael Shapiro. 'A note on context-sensitive languages and word problems'. In: *Internat. J. Algebra Comput.* 4.4 (1994), pp. 493–497. ISSN: 0218-1967. DOI: 10.1142/S0218196794000129.

[90] Michael Shapiro. 'Pascal's triangles in abelian and hyperbolic groups'. In: *J. Austral. Math. Soc. Ser. A* 63.2 (1997), pp. 281–288. ISSN: 0263-6115.

[91] Said Sidki. 'Automorphisms of one-rooted trees: growth, circuit structure, and acyclicity'. In: *J. Math. Sci. (New York)* 100.1 (2000). Algebra, 12, pp. 1925–1943. ISSN: 1072-3374. DOI: 10.1007/BF02677504.

[92] Michael Sipser. *Introduction to the Theory of Computation*. 3rd ed. Cengage Learning, 2013. ISBN: 9781133187790.

[93] A. M. Turing. 'On Computable Numbers, with an Application to the Entscheidungsproblem'. In: *Proc. London Math. Soc. (2)* 42.3 (1936), pp. 230–265. ISSN: 0024-6115. DOI: 10.1112/plms/s2-42.1.230.

[94] Xiaofeng Wang, Guo Li, Ling Yang and Hanling Lin. 'Groups with two generators having unsolvable word problem and presentations of Mihailova subgroups of braid groups'. In: *Comm. Algebra* 44.7 (2016), pp. 3020–3037. ISSN: 0092-7872. DOI: 10.1080/00927872.2015.1065867.

[95] Herbert S Wilf. *generatingfunctionology*. 2nd ed. Boston, MA: Academic Press Inc, 1994. URL: https://www2.math.upenn.edu/~wilf/gfology2.pdf.