2-15-2018

Higher rank lamplighter groups are graph automatic

Sophie Bérubé
Johns Hopkins Bloomberg School of Public Health

Tara Palnitkar
University of Minnesota Twin Cities

Jennifer Taback
Bowdoin College

Journal of Algebra 496 (2018) 315–343

Contents lists available at ScienceDirect

Journal of Algebra

www.elsevier.com/locate/jalgebra

Higher rank lamplighter groups are graph automatic ☆

Sophie Bérubé [a,1], Tara Palnitkar [b,1], Jennifer Taback [c,*,1]

[a] *Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, Baltimore, MD 21205, United States*
[b] *Department of Mathematics, University of Minnesota, Minneapolis, MN 55455, United States*
[c] *Department of Mathematics, Bowdoin College, Brunswick, ME 04011, United States*

A R T I C L E   I N F O

Article history:
Received 13 June 2017
Available online 6 November 2017
Communicated by Derek Holt

MSC:
20F65
68Q45

Keywords:
Automatic group
(Cayley) graph automatic group
Higher rank lamplighter group
Diestel–Leader group

A B S T R A C T

We show that the higher rank lamplighter groups, or Diestel–Leader groups $\Gamma_d(q)$ for $d \geq 3$, are graph automatic. This introduces a new family of graph automatic groups which are not automatic.

☆ The third author acknowledges support from National Science Foundation grant DMS-1105407 and Simons Foundation grant 31736 to Bowdoin College.

* Corresponding author.
E-mail address: jtaback@bowdoin.edu (J. Taback).

[1] The authors would like to thank Murray Elder for helpful conversations during the writing of this paper and the anonymous referee for detailed comments.

https://doi.org/10.1016/j.jalgebra.2017.09.038
0021-8693/© 2017 Elsevier Inc. All rights reserved.

1. Introduction

Automatic groups were formally introduced in [7] by Epstein, Cannon, Holt, Levy, Paterson and Thurston, motivated by initial observations of Cannon and Thurston about hyperbolic groups and their geometry. Their goal was first to understand fundamental groups of compact 3-manifolds, and then to streamline computation in these groups. For example, if G is an automatic group, then its Dehn function is at most quadratic, its word problem can be solved in quadratic time, and the automatic structure can be used to reduce any word in the generating set to a normal form for that group element, also in quadratic time.

A finitely generated group G has an automatic structure with respect to a generating set S if there is a regular language of normal forms for elements of G, and, for each $s \in S$, a finite state automaton which recognizes multiplication by s. The class of automatic groups includes all finite groups, braid groups, Coxeter groups, hyperbolic groups and mapping class groups, among others. It is shown in Section 2.4 of [7] that if G has an automatic structure with respect to one generating set, then it has an automatic structure with respect to any generating set. All automatic groups are finitely presented, and there are geometric conditions which can be used to show that a set of normal forms constitutes the basis of an automatic structure. For a comprehensive introduction to automatic groups, see [7], or for a shorter treatment, [9], [11] or [14].

It is unsatisfying that many groups which have nice algorithmic properties, including the properties listed above, are not automatic. For instance, a finitely generated nilpotent group is automatic if and only if it is virtually abelian. In [13], Kharlampovich, Khoussainov, and Miasnikov extend the definition of an automatic group to a *(Cayley) graph automatic group*, in which the language of normal forms representing group elements is defined over a finite alphabet of symbols. If one takes the symbol alphabet to be the generating set for the group, then the definition of an automatic group is recovered. Graph automatic groups retain many of the computational advantages of automatic groups, and this enlarged class includes the solvable Baumslag–Solitar groups $BS(1, n)$, the lamplighter groups $\mathbb{Z}_n \wr \mathbb{Z}$, the metabelian groups $\mathbb{Z}^n \ltimes_A \mathbb{Z}$ for $A \in SL_n(\mathbb{Z})$, and all finitely generated groups of nilpotency class at most 2, among others. [13] It is shown in [2] that the non-solvable Baumslag–Solitar groups $BS(m, n)$ are also graph automatic.

In this paper, we prove the following theorem.

Theorem 1. *The Diestel–Leader groups $\Gamma_d(q)$ for $d \geq 3$ are graph automatic.*

The family of Diestel–Leader groups $\Gamma_d(q)$ for $d \geq 3$, or higher rank lamplighter groups, was introduced in [1] by Bartholdi, Neuhauser and Woess. These groups are not automatic, as they are type F_{d-1} but not F_d when $d \geq 3$, and automatic groups are of type FP_∞. These groups are defined explicitly in Section 2 below, and their metric properties were studied by the third author and Stein in [16]. Kevin Wortman has sketched a proof showing that arguments analogous to those of Gromov in [10] imply that the Dehn function of $\Gamma_d(q)$ is quadratic regardless of the values of $d \geq 3$ and q. It was shown in [3] that when p is prime, $\Gamma_3(p)$ is a cocompact lattice in $Sol_5(\ \ _p((t)))$, and its Dehn function is quadratic. The Dehn function of $\Gamma_3(m)$ is studied for any m in [12] where it is shown to be at most quartic.

The Cayley graph of the Diestel–Leader group $\Gamma_d(q)$, with respect to a certain generating set is a *Diestel–Leader graph*, a particular subset of a product of d infinite trees of valence $q + 1$. More general Diestel–Leader graphs were introduced in [4] as a potential answer to the question "Is any connected, locally finite, vertex transitive graph quasi-isometric to the Cayley graph of a finitely generated group?" The Diestel–Leader graph which is a subset of a product of two infinite trees of differing valence is not quasi-isometric to the Cayley graph of any finitely generated group, as shown by Eskin, Fisher and Whyte in [8]. The Cayley graph of the well-known lamplighter group $L_q = \Gamma_2(q) = \mathbb{Z}_q \wr \mathbb{Z}$, with respect to a natural generating set, is the Diestel–Leader graph contained in a product of two infinite trees of valence $q + 1$. In this sense we view the Diestel–Leader groups as a geometric higher rank generalization of the lamplighter groups.

2. Diestel–Leader groups

We briefly introduce the Diestel–Leader groups $\Gamma_d(q)$ and their geometry, and refer the reader to [1], [16] and [17] for a more comprehensive treatment. The Diestel–Leader graph $DL_d(q)$ is the subset of the product of d infinite regular trees $T_1, T_2, \cdots, T_d$, each with valence $q + 1$ and a height function $h_i : T_i \to \mathbb{R}$, consisting of the vertices for which the sum of the heights of the coordinates is equal to zero. Two vertices are connected by an edge if and only if they are identical in all but two coordinates, and in those two coordinates, say i and j, the entries differ by an edge in both T_i and T_j.

Bartholdi, Neuhauser and Woess in [1] present a matrix group $\Gamma_d(q)$ with a particular generating set $S_d(q)$ so that the Cayley graph $\Gamma(\Gamma_d(q), S_d(q))$ is exactly the Diestel–Leader graph $DL_d(q)$. Their construction relies on the arithmetic condition that $d-1 < p$ for all prime divisors p of q. Specifically, let $\mathcal{L}_q$ be a commutative ring of order q with multiplicative unit 1, and suppose $\mathcal{L}_q$ contains distinct elements $l_1, \ldots, l_{d-1}$ such that if $d \geq 3$, their pairwise differences are invertible. In this paper, we additionally assume that l_i is also invertible, for $1 \leq i \leq d - 1$. These conditions are easily satisfied, for example, in $\mathbb{Z}_q$ for large enough q.

Define a ring of polynomials in the formal variables t and $(t + l_i)^{-1}$ for $1 \leq i \leq d - 1$ with finitely many nonzero coefficients lying in $\mathcal{L}_q$:

$$\mathcal{R}_d(\mathcal{L}_q) = \mathcal{L}_q[t, (t + l_1)^{-1}, (t + l_2)^{-1}, \cdots, (t + l_{d-1})^{-1}].$$

The Diestel–Leader group constructed in [1] is the group of affine matrices of the form:

$$\begin{pmatrix} (t + l_1)^{m_1} \cdots (t + l_{d-1})^{m_{d-1}} & P \\ 0 & 1 \end{pmatrix} \tag{1}$$

with $m_1, m_2, \cdots, m_{d-1} \in \mathbb{Z}$ and $P \in \mathcal{R}_d(\mathcal{L}_q)$, which has Cayley graph $DL_d(q)$ with respect to the generating set $S_d(q)$ consisting of the matrices

$$\begin{pmatrix} t+l_i & b \\ 0 & 1 \end{pmatrix}^{\pm 1}, \text{ with } b \in \mathcal{L}_q,\ i \in \{1, 2, \cdots, d-1\} \text{ and}$$

$$\begin{pmatrix} (t+l_i)(t+l_j)^{-1} & b(t+l_j)^{-1} \\ 0 & 1 \end{pmatrix}, \text{ with } b \in \mathcal{L}_q,\ i, j \in \{1, 2, \cdots, d-1\},\ i \neq j.$$

We refer to a matrix of the form $\begin{pmatrix} t+l_i & b \\ 0 & 1 \end{pmatrix}$ as a *type 1* generator and a matrix of the form $\begin{pmatrix} (t+l_i)(t+l_j)^{-1} & b(t+l_j)^{-1} \\ 0 & 1 \end{pmatrix}$ as a *type 2* generator. We assume that in a type 2 generator s we have $i < j$. If this is not the case, replace s by s^{-1} which reverses the roles of i and j.

An element $g \in \Gamma_d(q)$ is uniquely defined by a $(d-1)$-tuple $(m_1, m_2, \cdots, m_{d-1})$ of integers which determine the upper left entry of g and a polynomial $P \in \mathcal{R}_d(\mathcal{L}_q)$. The identification between a group element and a vertex in the Diestel–Leader graph $DL_d(q)$ is based on this information as well, and is explicitly described in [1] as well as [15], the extended version of [17]. Roughly, the tree T_i is associated with the variable $(t+l_i)^{-1}$, for $1 \leq i \leq d-1$, and T_d is associated with the variable t. Vertices in the tree T_i are assigned equivalence classes of Laurent polynomials in the corresponding variable. For $P \in \mathcal{R}_d(\mathcal{L}_q)$ we let $\mathcal{L}S_i(P)$ denote the Laurent polynomial obtained when P is rewritten in terms of the variable $t+l_i$ for $1 \leq i \leq d-1$, or t^{-1} when $i = d$. To find the coordinate in the tree T_i of the vertex corresponding to $g \in \Gamma_d(q)$, compute the Laurent polynomial

$$\mathcal{L}S_i((t+l_1)^{-m_1} \cdots (t+l_{d-1})^{-m_{d-1}} P)$$

and consider only those terms of negative degree, or nonpositive degree when $i = d$. This Laurent polynomial, along with the integer m_i (or $m_1 + \cdots + m_{d-1}$ when $i = d$) determines an equivalence class of polynomials associated to a vertex in T_i. To show this is well defined we refer to the following lemma, proven in [17]. The proof relies on rewriting $Q \in \mathcal{R}_d(\mathcal{L}_q)$ as a Laurent polynomial in each of the possible variables.

Lemma 2 *(Decomposition Lemma). Let*

$$Q \in \mathcal{R}_d(\mathcal{L}_q) = \mathcal{L}_q[(t+l_1)^{-1}, (t+l_2)^{-1}, \cdots, (t+l_{d-1})^{-1}, t].$$

Then Q can be written uniquely as $P_1(Q) + P_2(Q) + \cdots + P_d(Q)$ where

(a) *for $1 \leq i \leq d-1$ we have $P_i(Q) \in \mathcal{L}_q[(t+l_i)^{-1}]$ with constant term 0, and*
(b) *for $i = d$ we have $P_d(Q) \in \mathcal{L}_q[t]$.*

While we will not explicitly use the identification between a group element and the corresponding vertex in the Cayley graph in this paper, we will use the Decomposition Lemma repeatedly. That is, for $g \in \Gamma_d(q)$ as in Equation (1), we will decompose a polynomial related to P using the Decomposition Lemma, and use the component polynomials as the basis of our graph automatic structure. In our analysis of the relationship between g and gs, for $s \in S_d(q)$, we expand the upper right entry of the product gs using the same techniques.

It is easily verified that the following combinatorial formulae allow us to rewrite polynomials in $t + l_u$ in terms of $t + l_v$ for $u \neq v$, and t^{-1}:

$$(t + l_u)^k = \sum_{n=0}^{\infty} \binom{k}{n} (l_u - l_v)^{k-n} (t + l_v)^n \tag{2}$$

and

$$(t + l_u)^k = \sum_{n=-k}^{\infty} \binom{k}{-n} l_v^{n+k} t^{-n} \tag{3}$$

for any $k \in \mathbb{Z}$. Moreover, when k is nonnegative, we write t^k as a polynomial in $t + l_v$ as follows:

$$t^k = \sum_{n=0}^{k} \binom{k}{n} (-l_v)^{k-n} (t + l_v)^n. \tag{4}$$

In the proofs below, we repeatedly use Equations (2) and (3) with a fixed value of $u \in \{1, 2, \cdots, d-1\}$ and $k = -1$ to rewrite $(t+l_u)^{-1}$ in terms of $t+l_v$, for $u \neq v$, or t^{-1}. In the first case, we have:

$$(t + l_u)^{-1} = \sum_{n=0}^{\infty} \alpha_n (t + l_v)^n \tag{5}$$

where $\alpha_n = \binom{-1}{n}(l_u - l_v)^{-(n+1)} = (-1)^n (l_u - l_v)^{-(n+1)}$. Writing $C_{u,v} = (l_u - l_v)^{-1}$ for $1 \leq v \leq d-1$, this simplifies to $\alpha_n = (-1)^n C_{u,v}^{n+1}$. Notice that for fixed values of u and v, we have $\alpha_{n+1} = -C_{u,v}\alpha_n$.

Similarly, we simplify Equation (3) with exponent $k = -1$ as follows:

$$(t + l_u)^{-1} = \sum_{n=1}^{\infty} \binom{-1}{-n} (l_u)^{n-1} t^{-n} = \sum_{n=1}^{\infty} (-1)^{n-1} (l_u)^{n-1} t^{-n}. \tag{6}$$

Denote the coefficients in the above sum as $\alpha'_n = (-1)^{n-1}(l_u)^{n-1}$, and note that $\alpha'_{n+1} = -l_u \alpha'_n$.

We make one assumption about our Diestel–Leader groups to simplify notation throughout this paper. Namely we take $\mathcal{L}_q = \mathbb{Z}_q$ and note that all our theorems hold for more general coefficient rings as well.

3. Graph automatic groups

Let G be a group with finite symmetric generating set X, and Λ a finite set of symbols. The number of symbols (letters) in a word $u \in \Lambda^*$ is denoted $|u|_\Lambda$. We begin by defining a convolution of group elements, following [13] in our notation.

Definition 1 *(Convolution).* Let Λ be a finite set of symbols, $\diamond$ a symbol not in Λ, and let $L_1, \ldots, L_k$ be a finite set of languages over Λ. Set $\Lambda_\diamond = \Lambda \cup \{\diamond\}$. Define the *convolution of a tuple* $(w_1, \ldots, w_k) \in L_1 \times \cdots \times L_k$ to be the string $\otimes(w_1, \ldots, w_k)$ of length $\max |w_i|_\Lambda$ over the alphabet $(\Lambda_\diamond)^k$ as follows. The i-th symbol of the string is

$$\begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_k \end{pmatrix}$$

where λ_j is the ith letter of w_j if $i \leq |w_j|_\Lambda$ and $\diamond$ otherwise. Then

$$\otimes(L_1, \ldots, L_k) = \{\otimes(w_1, \ldots, w_k) \mid w_i \in L_i\}.$$

We note that the convolution of regular languages is again a regular language. When $L_i = \Lambda^*$ for all i the exact definition in [13] is recovered.

As an example, if $w_1 = abb, w_2 = bbb$ and $w_3 = ba$ then

$$\otimes(w_1, w_2, w_3) = \begin{pmatrix} a \\ b \\ b \end{pmatrix} \begin{pmatrix} b \\ b \\ a \end{pmatrix} \begin{pmatrix} b \\ b \\ \diamond \end{pmatrix}$$

The definition of a graph automatic group extends that of an automatic group by allowing the normal forms for group elements to be defined over a finite alphabet of symbols. When this set of symbols is simply taken to be the set of group generators, the definition of an automatic group is recovered.

A set of normal forms for a group may additionally be quasi-geodesic.

Definition 2 *(Quasigeodesic normal form).* A *normal form for* (G, X, Λ) is a set of words $L \subseteq \Lambda^*$ in bijection with G. A normal form L is *quasigeodesic* if there is a constant D so that

$$|u|_\Lambda \leq D(||u||_X + 1)$$

for each $u \in L$, where $||u||_X$ is the length of a geodesic in X^* for the group element represented by u.

The $||u||_X + 1$ in the definition allows for normal forms where the identity of the group is represented by a nonempty string of length at most D. We denote the image of $u \in L$ under the bijection with G by $\overline{u}$.

The following definition was introduced in [13].

Definition 3 *(Graph automatic group).* Let (G, X) be a group and finite symmetric generating set, and Λ a finite set of symbols. We say that (G, X, Λ) is *graph automatic* if there is a regular normal form $L \subset \Lambda^*$, such that for each $x \in X$ the language $L_x = \{\otimes(u, v) \mid u, v \in L, \overline{v} =_G \overline{u}x\}$ is a regular language.

The language L_x is often referred to as a *multiplier language.*

Any set of normal forms forming the basis of an automatic structure for a group G is automatically quasigeodesic. The proof of Lemma 8.2 of [13] contains the observation that graph automatic groups naturally possess a quasigeodesic normal form, and a proof is included in [5].

Lemma 3. *Let G be a group with finite generating set X. If (G, X, Λ) is graph automatic with respect to the regular normal form L, then L is a quasigeodesic normal form.*

The existence of a quasigeodesic regular normal form in an automatic or graph automatic structure ensures that the word problem is solvable in quadratic time. While the Diestel–Leader groups are metabelian, and hence have solvable word problem, we note that it is a direct consequence of Theorem 9 of [16] that the set of normal forms defined in Section 4 is quasi-geodesic.

We conclude with two straightforward lemmas about convolutional languages which we will refer to in the verification of our graph automatic structure for $\Gamma_d(q)$.

Lemma 4 *(Offset Lemma). Let $\mathcal{A}$ and $\mathcal{B}$ be regular languages, with*

$$\mathcal{A}' = \{\otimes(a_1, a_2) | a_i \in \mathcal{A}, |a_1| = |a_2|\}$$

and $\mathcal{B}'$ any subset of $\otimes(\mathcal{B}, \mathcal{B})$ which is a regular language. Let Λ be any finite alphabet. Then

$$\{\otimes(a_1b_1, a_2xb_2), \otimes(a_1yb_1, a_2b_2) | \otimes (a_1, a_2) \in \mathcal{A}', \otimes(b_1, b_2) \in \mathcal{B}', x, y \in \Lambda\}$$

is a regular language.

The proof of Lemma 4 follows easily from the next lemma, whose proof is given in [18].

Lemma 5. *Let $\mathcal{L}$ be a regular language defined over a finite alphabet Λ. Then the set*

$$\{\otimes(xw, w) | w \in \mathcal{L},\ x \in \Lambda\}$$

forms a regular language.

4. A regular language of normal forms

We begin the construction of a graph automatic structure for $\Gamma_d(q)$ with a quasi-geodesic normal form. Let $g = \begin{pmatrix} \Pi_{k=1}^{d-1}(t+l_k)^{m_k} & R \\ 0 & 1 \end{pmatrix}$. The vector of integers $\mathbf{m} = (m_1, m_2, \cdots, m_{d-1})$ and the polynomial R uniquely define g. However, g is also uniquely defined from the data $\mathbf{m}$ and the polynomial

$$R' = \Pi_{m=1}^{d-1}(t+l_i)^{-m_i} R$$

and this forms the basis of the normal form we use for our regular language. Namely, using the Decomposition Lemma from [17] (Lemma 2.1 in [17] and Lemma 2 in this paper) we can uniquely decompose R' as follows:

$$R' = R_1 + R_2 + \cdots R_d$$

where R_i for $1 \leq i < d$ contains only terms in the variable $t + l_i$ of negative degree, and R_d contains terms in the variable t^{-1} of nonpositive degree.

As concatenations of regular languages are regular, we define a prefix language and a suffix language for our normal form which are both regular, and whose union gives a regular language of normal forms for elements of $\Gamma_d(q)$. Let the prefix language $\mathcal{P}$ be defined over the alphabet $\{x, y, \#\}$, and encode the vector $(m_1, m_2, \cdots, m_{d-1})$ as $\epsilon_1^{m_1}\#\epsilon_2^{m_2}\#\cdots\#\epsilon_{d-1}^{m_{d-1}}$, where $\epsilon_i = x$ if $m_i > 0$ and $\epsilon_i = y$ if $m_i < 0$; if $m_i = 0$ we omit ϵ_i. Note that the prefix corresponding to the identity is $\#^{d-2}$. The collection of these strings forms a regular language which we denote $\mathcal{P}$.

We now encode the information contained in the polynomials $R_1, \cdots, R_d$ as a suffix language $\mathcal{S}$ over the finite alphabet $\{\#, b_0, b_1, \cdots, b_{q-1}\}$ where $\{b_0, b_1, \cdots, b_{q-1}\} = \mathbb{Z}_q$. If $R_i \neq 0$, denote its minimal degree $-\delta_i$, for $\delta_i \in \mathbb{N}$, and when $i \neq d$ write

$$R_i = \sum_{j=1}^{\delta_i} \beta_{i,j}(t+l_i)^{-j}.$$

When $i = d$ we include a constant term in the polynomial expression, which is written in the variable t.

Let $S_i = \beta_{i,1}\beta_{i,2}\cdots\beta_{i,\delta_i}$ with $\beta_{i,j} \in \mathbb{Z}_q$ for $1 \leq i \leq d$ denote the string of coefficients of R_i, with $S_i = \emptyset$ if $R_i = 0$. The entry in the suffix language $\mathcal{S}$ corresponding to the tuple $R_1, \cdots, R_d$ is

$$S_1\#S_2\#\cdots\#S_d.$$

We use the string $\#^{d-1}$ to denote the suffix string when we have $R_k = 0$ for $1 \leq k \leq d$. Let $\mathcal{S}$ be the union of all strings of the above form; it is clear that $\mathcal{S}$ is a regular language and hence the language of concatenations $\mathcal{N} = \mathcal{P}\mathcal{S}$ describes a regular language of normal forms for elements of $\Gamma_d(q)$. Elements of the normal form language $\mathcal{N}$ will be written as (p, s) for some $p \in \mathcal{P}$ and $s \in \mathcal{S}$. If $u \in \mathcal{N}$, let $\bar{u}$ denote the corresponding element of $\Gamma_d(q)$, and if $g \in \Gamma_d(q)$ we let $\nu(g)$ denote the corresponding element of $\mathcal{N}$, with $\pi(g)$ and $\sigma(g)$, respectively, denoting the prefix and suffix strings of $\nu(g)$.

5. Multiplier languages

We now show that the multiplier languages $\mathcal{L}_s$, where $s \in S_d(q)$, arising from this normal form $\mathcal{N}$ are also regular. The multiplier language $\mathcal{L}_s$ consists of convolutions $\otimes(\nu(g), \nu(gs))$ where $g \in \Gamma_d(q)$. It suffices to check that $\mathcal{L}_s$ is regular when s has one of two forms:

$$s \in \left\{ \begin{pmatrix} t+l_i & b \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} (t+l_i)(t+l_j)^{-1} & b(t+l_j)^{-1} \\ 0 & 1 \end{pmatrix} \right\}$$

with $b \in \mathbb{Z}_q$, as it is proven in [6] that $\mathcal{L}_s$ is a regular language if and only if $\mathcal{L}_{s^{-1}}$ is a regular language. Recall that when s is a type 2 generator, we assume that $i < j$. The following theorem constructs the remainder of the graph automatic structure for $\Gamma_d(q)$.

Theorem 6. *Let $s \in S_d(q)$ be a*

- *a type 1 generator of $\Gamma_d(q)$ of the form $\begin{pmatrix} t+l_i & b \\ 0 & 1 \end{pmatrix}$, or*
- *a type 2 generator of the form $\begin{pmatrix} (t+l_i)(t+l_j)^{-1} & b(t+l_j)^{-1} \\ 0 & 1 \end{pmatrix}$*

where $1 \leq i, j \leq d-1$, $i < j$ and $b \in \mathbb{Z}_q$. The language

$$\mathcal{L}_s = \{\otimes(\nu(g), \nu(gs)) | g \in \Gamma_d(q)\}$$

is a regular language.

Fix a generator s, which necessarily fixes a value of i and possibly also of j. We prove Theorem 6 in several steps. First observe that the relationship between the prefix strings $\pi(g) = (m_1, m_2, \cdots, m_{d-1})$ and $\pi(gs) = (m'_1, m'_2, \cdots, m'_{d-1})$ is easily determined:

1. If s is a type 1 generator, $m'_i = m_i + 1$ and $m'_k = m_k$ for all other k.
2. If s is a type 2 generator, then $m'_i = m_i + 1$ and $m'_j = m_j - 1$. For all $k \neq i, j$ we have $m'_k = m_k$.

These conditions are easily checked with a finite state machine, and we conclude that for a fixed generator, the set of strings $\mathcal{P}_s = \{\otimes(p_1, p_2)|p_i \in \mathcal{P}\}$ satisfying the appropriate condition is a regular language.

The next step in the proof of Theorem 6 is to determine the relationship between $\sigma(g)$ and $\sigma(gs)$, and construct a finite state machine which recognizes this relationship between any two suffix strings. Beginning with a pair of elements $g, gs \in \Gamma_d(q)$:

- let the polynomials $R_1, R_2, \cdots R_d$ determine $\sigma(g)$ and $Q_1, Q_1, \cdots, Q_d$ determine $\sigma(gs)$;
- the strings of coefficients of R_k and Q_k are denoted S_k and S'_k, respectively;
- the suffix strings $\sigma(g)$ and $\sigma(gs)$ are then written, respectively, as $\sigma(g) = S_1\#S_2\#\cdots\#S_d$ and $\sigma(gs) = S'_1\#S'_2\#\cdots\#S'_d$.

In Sections 6 and 7 we determine the algebraic relationship between the entries of S_k and S'_k, and build a finite state machine which recognizes this relationship. Namely, for a fixed generator s we construct finite state automata which verify:

1. For each $n \neq i$, that the strings S_n and S'_n differ in the appropriate manner. This step creates $d-1$ finite state automata.
2. When $n = i$, that the first entry in S'_i relates to the remaining coefficients in $\sigma(g)$ in the appropriate manner. This entry corresponds to the coefficient of $(t+l_i)^{-1}$ in Q_i.
3. When $n = i$ that the remaining entries in S_i and S'_i differ in appropriate manner.

These finite state automata are combined to create an automaton M_s which accepts a regular language $\mathcal{N}'_s$ of convolutions $\otimes(\sigma(g), \sigma(h))$, for $g, h \in \Gamma_d(q)$. This language includes $\{\otimes(\sigma(g), \sigma(gs))|g \in \Gamma_d(q)\}$. Define $\mathcal{N}_s$ to be the language of concatenations $\mathcal{P}_s\mathcal{N}'_s$. The following proposition follows immediately.

Proposition 7. *The language of concatenations $\mathcal{N}_s = \mathcal{P}_s\mathcal{N}'_s$ is a regular language.*

To conclude the proof of Theorem 6 we show in Section 7 that $\mathcal{N}_s = \mathcal{L}_s$.

6. Construction of automata I

Let s be either a type 1 or type 2 generator, so values of i and possibly j are fixed. In this section we determine the relationship between the coefficients of R_n and Q_n arising from $\sigma(g)$ and $\sigma(gs)$, respectively, when $n \neq i$ and construct automata which recognize this relationship.

6.1. Analysis of coefficients for type 1 generators

Consider pairs $\otimes(\nu(g), \nu(gs))$ where s is a type 1 generator of the form $\begin{pmatrix} t+l_i & b \\ 0 & 1 \end{pmatrix}$. With $g = \begin{pmatrix} \Pi_{k=1}^{d-1}(t+l_k)^{m_k} & R \\ 0 & 1 \end{pmatrix}$ we compute

$$gs = \begin{pmatrix} (t+l_i)\Pi_{k=1}^{d-1}(t+l_k)^{m_k} & b\Pi_{k=1}^{d-1}(t+l_k)^{m_k} + R \\ 0 & 1 \end{pmatrix}.$$

Using the Decomposition Lemma (Lemma 2), write

$$\Pi_{k=1}^{d-1}(t+l_i)^{-m_k}R = R_1 + R_2 + \cdots + R_d \tag{7}$$

and

$$\begin{aligned} &(t+l_i)^{-1}\Pi_{k=1}^{d-1}(t+l_k)^{-m_k}(b\Pi_{k=1}^{d-1}(t+l_k)^{m_k} + R) \\ &\qquad = b(t+l_i)^{-1} + (t+l_i)^{-1}(R_1 + R_2 + \cdots + R_d) \qquad (8) \\ &\qquad = Q_1 + Q_2 + \cdots + Q_d \end{aligned}$$

where the latter decomposition into polynomials Q_k is also obtained via the Decomposition Lemma. We now use Laurent polynomials to describe the exact relationship between R_n and Q_n and show that any differences between them can be detected by a finite state machine. Namely, for $1 \le n \le d$ and $n \ne i$, we compute the Laurent polynomial $\mathcal{L}S_n$ of the left hand side of Equation (8) and then consider the terms of negative degree, which form Q_n for $n \ne d$, and when $n = d$ the terms of nonpositive degree, which form Q_d.

First note that $\mathcal{L}S_i(b(t+l_i)^{-1}) = b(t+l_i)^{-1}$ and when $n \ne i$ we see from Equations (5) and (6) that $\mathcal{L}S_n(b(t+l_i)^{-1})$ contains no terms of negative degree when $n \ne i, d$ and no terms of nonpositive degree when $n = d$. Thus $b(t+l_i)^{-1}$ will add a term to Q_i but no other Q_n for $n \ne i$.

Recall that i is fixed by our choice of generator s. Rewrite the left hand side of Equation (8) as a Laurent polynomial in $t + l_n$ if $1 \le n \le d-1$, and t^{-1} if $k = d$. When $k \ne n, d$ we see that when $(t+l_i)^{-1}R_k$ is rewritten as an expression in $t + l_n$ there are no terms of negative degree. Hence this polynomial contributes no terms to Q_n. To see this, recall that above we wrote

$$R_k = \sum_{z=1}^{\delta_k} \beta_{k,z}(t+l_k)^{-z}$$

where $-\delta_j$ is the minimal degree of R_k, and hence a generic term from $(t+l_i)^{-1}R_k$ has the form $\beta(t+l_i)^{-1}(t+l_k)^{-m}$ for some $\beta \in \mathbb{Z}_q$ and $m \in \mathbb{N}$. We rewrite this in terms of $t + l_n$ using Equations (2) and (5) as

$$\beta(t+l_i)^{-1}(t+l_k)^{-m} = \beta\left(\sum_{y=0}^{\infty}\alpha_y(t+l_n)^y\right)\left(\sum_{x=0}^{\infty}\xi_x(t+l_n)^x\right)$$

where the ξ_x and α_y are the coefficients computed in Equations (2) and (5), and note that there are no terms of negative degree.

When $k=d$, the above argument holds with $t+l_k$ replaced by t^{-1} and any application of Equation (2) replaced by Equation (4).

Thus it must be the case that the terms of $\mathcal{L}S_n((t+l_i)^{-1}R_n)$ of negative degree form the polynomial Q_n. If $R_n=0$ then $Q_n=0$ as well. First consider the case $n\neq i,d$. Write

$$R_n = \sum_{z=1}^{\delta_n}\beta_{n,z}(t+l_n)^{-z}$$

and it follows that

$$(t+l_i)^{-1}R_n = \left(\sum_{y=0}^{\infty}\alpha_y(t+l_n)^y\right)\left(\sum_{z=1}^{\delta_n}\beta_{n,z}(t+l_n)^{-z}\right)$$

where the α_y are computed in Equation (5).

To simplify notation in the following argument, write

$$\begin{aligned}(t+&l_i)^{-1}R_n\\ &= \left(\sum_{y\geq 0}\alpha_y(t+l_n)^y\right)\left(\beta_1(t+l_n)^{-1}+\beta_2(t+l_n)^{-2}+\cdots+\beta_{\delta_n}(t+l_n)^{-\delta_n}\right)\end{aligned} \tag{9}$$

where $\beta_z\in\mathbb{Z}_q$ and $\delta_n\neq 0,\ \beta_{\delta_n}\neq 0$.

We see that multiplying R_n by each term in the infinite sum produces a pattern in the resulting coefficients. As we multiply R_n successively by each term in the infinite sum above, we keep track of the resulting coefficients of the terms of negative degree of Q_n in Table 1.

Now compute the coefficient γ_k of $(t+l_n)^{-k}$ in Q_n by summing the entries of the appropriate column of Table 1 to obtain

$$\gamma_k = \sum_{y=0}^{\delta_n-k}\alpha_y\beta_{y+k} \tag{10}$$

for $1\leq k\leq\delta_n-1$, and $\gamma_{\delta_n}=C_{n,i}\beta_{\delta_n}$ since $\alpha_0=C_{n,i}$. Recall that $C_{n,i}=(l_i-l_n)^{-1}$ and hence is invertible. As i is fixed by our choice of generator, we shorten $C_{n,i}$ to C_n for the remainder of the paper.

Table 1
As the multiplication in Equation (9) is carried out, like terms are grouped together and we keep track of the resulting coefficients of the terms of negative degree in this table. Each coefficient γ_k of $(t + l_n)^{-k}$ in Q_n is the sum of the coefficients in its column.

Degree in Q_n	$(t+l_n)^{-1}$	$(t+l_n)^{-2}$	$(t+l_n)^{-3}$	$\cdots$	$\beta_{\delta_n-1}(t+l_n)^{-\delta_n+1}$	$\beta_{\delta_n}(t+l_n)^{-\delta_n}$
Mult R_n by α_0	$\alpha_0\beta_1$	$\alpha_0\beta_2$	$\alpha_0\beta_3$	$\cdots$	$\alpha_0\beta_{\delta_n-1}$	$\alpha_0\beta_{\delta_n}$
Mult R_n by $\alpha_1(t+l_n)$	$\alpha_1\beta_2$	$\alpha_1\beta_3$	$\alpha_1\beta_4$	$\cdots$	$\alpha_1\beta_{\delta_n}$	0
Mult R_n by $\alpha_2(t+l_n)^2$	$\alpha_2\beta_3$	$\alpha_2\beta_4$	$\alpha_2\beta_5$	$\cdots$	0	0
Mult R_n by $\alpha_3(t+l_n)^3$	$\alpha_3\beta_4$	$\alpha_3\beta_5$	$\alpha_3\beta_6$	$\cdots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Final Coefficients in Q_n	γ_1	γ_2	γ_3	$\cdots$	γ_{δ_n-1}	γ_{δ_n}

Notice that for $k < \delta_n - 1$ it follows from Equation (10) that

$$\gamma_k = -C_n\gamma_{k+1} + \alpha_0\beta_k = -C_n\gamma_{k+1} + C_n\beta_k$$

and hence

$$\gamma_{k+1} = -C_n^{-1}(\gamma_k - C_n\beta_k) = -C_n^{-1}\gamma_k + \beta_k.$$

In Section 6.3 we will use the expression of γ_{k+1} in terms of γ_k to construct a finite state machine which recognizes the relationship between the coefficients of R_n and Q_n. It follows from this equation that a pair (β_k, γ_k) of coefficients of $(t+l_n)^{-k}$ in R_n and Q_n, respectively, uniquely determine the coefficient γ_{k+1} of $(t + l_k)^{-(k+1)}$ in Q_n.

If we take $n = d$ and replace any instance of Equation (5) with Equation (6), and Equation (2) with Equation (3), we can make an analogous argument. In this case, R_d and Q_d include a constant term. We also rely on the fact that α'_k and α'_{k+1}, the coefficients in Equation (5), are related in the same way as α_k and α_{k+1}, the coefficients in Equation (2). In this argument we use the assumption that the l_j are chosen to be invertible. We include a few details for completeness.

When $n = d$, write

$$\begin{aligned}(t+l_i)^{-1}R_d & \\ &= \left(\sum_{y\geq 1}\alpha'_y(t^{-1})^y\right)\left(\beta_0 + \beta_1(t^{-1})^{-1} + \beta_2(t^{-1})^{-2} + \cdots + \beta_{\delta_d}(t^{-1})^{-\delta_d}\right)\end{aligned} \tag{11}$$

where $\beta_i \in \mathbb{Z}_q$ and $\beta_{\delta_d} \neq 0$. We construct a table analogous to that in Table 1, and obtain coefficients $\gamma_0, \gamma_1 \cdots \gamma_{\delta_d}$ for Q_d satisfying

$$\gamma_k = \sum_{y=1}^{\delta_d - k}\alpha'_y\beta_{y+k} \tag{12}$$

for $0 \leq k \leq \delta_d - 1$.

As when $n < d$, recall that $\alpha'_{y+1} = -l_i\alpha'_y$ and these coefficients satisfy the relationship

$$\gamma_k = -l_i\gamma_{k+1} + \alpha'_1\beta_{k+1}$$

and hence, as $\alpha'_1 = 1$,

$$\gamma_{k+1} = -l_i^{-1}(\gamma_k - \beta_{k+1}).$$

Notice that unlike the case of $n \neq i, d$, the coefficient γ_{k+1} depends both on γ_k and β_{k+1}. This will create different transition functions in the finite state machine constructed in Section 6.3 when $n = d$.

A proof of the following lemma is contained in the above exposition when $n \neq i$; it is proven through repeated application of Equations (2), (3), and (4) to the expression in Equation (8). The case $n = i$ is verified in Section 7. We state it for easy reference.

Lemma 8. *Fix a type 1 generator s, and let $g \in \Gamma_d(q)$. Using the decompositions above in Equations (7) and (8), we see that:*

1. *If $n \neq i, d$ then the minimal degree of R_n is the same as the minimal degree of Q_n.*
2. *If $n = d$ then the minimal degree of Q_i is one greater than the minimal degree of R_i.*
3. *If $n = i$ then the minimal degree of Q_i is one less than the minimal degree of R_i.*

6.2. Analysis of coefficients for type 2 generators

We now perform the same analysis on the coefficients of the polynomials which determine $\sigma(g)$ and $\sigma(gs)$ when s is a type 2 generator of the form

$$\begin{pmatrix} (t+l_i)(t+l_j)^{-1} & b(t+l_j)^{-1} \\ 0 & 1 \end{pmatrix} \text{ with } i < j.$$

If $g = \begin{pmatrix} \Pi_{k=1}^{d-1}(t+l_k)^{m_k} & R \\ 0 & 1 \end{pmatrix}$, compute

$$gs = \begin{pmatrix} (t+l_i)(t+l_j)^{-1}\Pi_{m=1}^{d-1}(t+l_i)^{m_i} & b(t+l_j)^{-1}\Pi_{m=1}^{d-1}(t+l_i)^{m_i} + R \\ 0 & 1 \end{pmatrix}.$$

Apply the Decomposition Lemma to write

$$\Pi_{m=1}^{d-1}(t+l_i)^{-m_i}R = R_1 + R_2 + \cdots + R_d$$

and compute

$$\begin{aligned} &(t+l_i)^{-1}(t+l_j)\Pi_{m=1}^{d-1}(t+l_i)^{-m_i}(b(t+l_j)^{-1}\Pi_{m=1}^{d-1}(t+l_i)^{m_i} + R) \\ &\quad = b(t+l_i)^{-1} + (t+l_i)^{-1}(t+l_j)R \\ &\quad = b(t+l_i)^{-1} + (t+l_i)^{-1}(t+l_j)(R_1 + R_2 + \cdots + R_d) \\ &\quad = Q_1 + Q_2 + \cdots Q_{d-1} + Q_d \end{aligned} \tag{13}$$

where the last line is obtained by applying the Decomposition Lemma to the original polynomial expression. We must compute Q_k and show in Section 6.3 that its relationship to R_k can be verified by a finite state machine.

Recall that when s is a type 2 generator, the values of $1 \leq i < j \leq d-1$ and $b \in \mathbb{Z}_q$ are fixed. We defined $C_n = (l_i - l_n)^{-1}$ and let $D_n = l_j - l_n$ for the fixed values of i and j.

First note that when $n \neq i, d$, we can write

$$b(t+l_i)^{-1} = b \sum_{y=0}^{\infty} \alpha_y (t+l_n)^y$$

where α_y is computed in Equation (5). As this expression has no terms of negative degree, we do not need to consider $b(t+l_i)^{-1}$ when computing Q_n for $n \neq i, d$. When $n = d$ we use Equation (6) in place of Equation (5) and observe that there are no terms of nonpositive degree in $\mathcal{LS}_d(b(t+l_i)^{-1})$. Hence $b(t+l_i)^{-1}$ does not play a role in determining Q_d.

Rewrite $(t+l_i)^{-1}(t+l_j)$ as an expression in the variable $t+l_n$ for $n \neq i, j, d$ as

$$\begin{aligned} \mathcal{LS}_n((t+l_i)^{-1}(t+l_j)) &= \left(\sum_{y=0}^{\infty} \alpha_y (t+l_n)^y\right)((l_j - l_n) + (t+l_n)) \\ &= \left(\sum_{y=0}^{\infty} \alpha_y (t+l_n)^y\right)(D_n + (t+l_n)) \\ &= C_n D_n + \sum_{y=1}^{\infty} (D_n \alpha_y + \alpha_{y-1})(t+l_n)^y \\ &= \sum_{y=0}^{\infty} \sigma_y (t+l_n)^y \end{aligned} \tag{14}$$

where $\sigma_0 = C_n D_n$, for $y > 1$ we have $\sigma_y = D_n \alpha_y + \alpha_{y-1}$ and α_y is computed as in Equation (5). Since $\alpha_{y+1} = -C_n \alpha_y$, it follows that for $y \geq 1$ we have $\sigma_{y+1} = -C_n \sigma_y$.

When $n = j$ the above expression simplifies to

$$\mathcal{LS}_j((t+l_i)^{-1}(t+l_j)) = \sum_{y=1}^{\infty} \alpha_{y-1} (t+l_j)^y. \tag{15}$$

When $n = d$ we obtain

$$\begin{aligned} \mathcal{LS}_d((t+l_i)^{-1}(t+l_j)) &= \left(\sum_{r=1}^{\infty} (-l_i)^{r-1} (t^{-1})^r\right)((t^{-1})^{-1} + l_j) \\ &= \sum_{r=0}^{\infty} \tau_r (t^{-1})^r \end{aligned} \tag{16}$$

where $\tau_0 = 1$ and for $r > 0$ we have $\tau_r = (-1)^{r-1}(-l_i)^{r-1}(-l_i + l_j)$.

For the remainder of this section, we assume that $n \neq i$. Now we compute $\mathcal{L}S_n((t+l_i)^{-1}(t+l_j)R_k)$ and show that when $k \neq n$ there are no terms of negative (resp. nonpositive when $n = d$) degree, and hence the terms of Q_n are exactly the terms of $\mathcal{L}S_n((t+l_i)^{-1}(t+l_j)R_n)$ of negative (resp. nonpositive) degree.

A generic term in $(t+l_i)^{-1}(t+l_j)R_k$ has the form

$$(t+l_i)^{-1}(t+l_j)\xi(t+l_k)^{-e},$$

with $\xi \in \mathbb{Z}_q$ and $e \in \mathbb{N}$. When $n \neq i, j, d$ and $k \neq d$,

$$\begin{aligned}&\mathcal{L}S_n((t+l_i)^{-1}(t+l_j)\xi(t+l_k)^{-e})\\ &\qquad = \begin{cases} \sum_{x=0}^{\infty} \sigma_x (t+l_n)^x \xi \sum_{z=0}^{\infty} \chi_z (t+l_n)^z & \text{when } n \neq j, d \\ \sum_{x=1}^{\infty} \alpha_{x-1} (t+l_j)^x \xi \sum_{z=0}^{\infty} \chi_z (t+l_j)^z & \text{when } n = j \end{cases}\end{aligned}$$

where Equations (2), (14) and (15) are used to obtain these expressions, neither of which contains any terms of negative degree in the variable $t + l_n$.

When $n \neq i, j, d$ and $k = d$ we expand a generic term of $(t+l_i)^{-1}(t+l_j)R_d$ as

$$\mathcal{L}S_n((t+l_i)^{-1}(t+l_j)\xi(t^{-1})^{-e}) = \sum_{x=0}^{\infty} \sigma_x (t+l_n)^x \xi \sum_{r=0}^{e} \binom{e}{r} (-l_n)^{e-r} (t+l_n)^r \qquad (17)$$

which has no terms of negative degree. In the equation above, σ_x is defined in Equation (14), $\xi \in \mathbb{Z}_q$ is the coefficient of $(t^{-1})^{-e}$ in R_d, and $e \in \mathbb{N}$. Thus Q_n is comprised of the terms of $\mathcal{L}S_n((t+l_i)^{-1}(t+l_j)R_n)$ of negative degree.

When $n = d$, we use Equations (3) and (6) in place of Equations (2) and (5), and Equation (16) to obtain the expression

$$\mathcal{L}S_d((t+l_i)^{-1}(t+l_j)\xi(t+l_k)^{-e}) = \sum_{x=0}^{\infty} \tau_x (t^{-1})^x \xi \sum_{r=e}^{\infty} \binom{e}{-r} (l_i)^{r+e} (t^{-1})^r$$

and since $e \geq 1$, this has no terms of nonpositive degree. Thus the terms of Q_d are exactly the terms of $\mathcal{L}S_d((t+l_i)^{-1}(t+l_j)R_d$ of nonpositive degree.

Our computations now mimic those in Section 6.1. When $n \neq i, j, d$, we can express $\mathcal{L}S_n((t+l_i)^{-1}(t+l_j)R_n)$ as

$$\sum_{x=0}^{\infty} \sigma_x (t+l_n)^x \left(\beta_1 (t+l_n)^{-1} + \beta_2 (t+l_n)^{-2} + \cdots + \beta_{\delta_n} (t+l_n)^{-\delta_n}\right)$$

where σ_x is defined in Equation (14), and create a chart analogous to Table 1. This allows us to compute the coefficients γ_k in Q_n as

$$\gamma_k = \sum_{y=0}^{\delta_n - k} \sigma_y \beta_{y+k} \tag{18}$$

where σ_y is defined in Equation (14) and $\sigma_0 = C_n D_n$. Recall that for $y \geq 1$ we have $\sigma_{y+1} = -C_n \sigma_y$. Hence, as in the case of type 1 generators, we see that

$$\gamma_k = -C_n \gamma_{k+1} + \sigma_0 \beta_k = -C_n \gamma_{k+1} + C_n D_n \beta_k.$$

When $n = j$, we begin with Equation (15) and express the Laurent polynomial $\mathcal{L}S_j((t+l_i)^{-1}(t+l_j)R_j$ as

$$\sum_{y=1}^{\infty} \alpha_{y-1}(t+l_j)^y \left(\beta_{j,1}(t+l_j)^{-1} + \beta_{j,2}(t+l_j)^{-2} + \cdots + \beta_{j,\delta_n}(t+l_j)^{-\delta_j}\right)$$

from which we construct a chart for the coefficients of Q_j and determine that

$$\gamma_k = \sum_{y=0}^{\delta_j - k} \alpha_y \beta_{y+k+1}.$$

Reasoning analogous to previous cases yields

$$\gamma_{k+1} = -C_j^{-1}(\gamma_k + \beta_{k+1}).$$

When $n = d$, replace instances of Equations (2) and (5) with Equations (3) and (6) to see that Q_d is the sum of the terms of $\mathcal{L}S_d((t+l_i)^{-1}(t+l_j)R_d)$ of nonpositive degree. To compute this, we write

$$\begin{aligned}
&(t+l_i)^{-1}(t+l_j)(\beta_0 + \beta_1(t^{-1})^{-1} + \beta_2(t^{-1})^{-2} + \cdots + \beta_{\delta_d}(t^{-1})^{-\delta_d}) \\
&= \left(\sum_{x=1}^{\infty}(-1)^{x-1} l_i^{x-1}(t^{-1})^x\right)((t^{-1})^{-1} + l_j)(\beta_0 + \beta_1(t^{-1})^{-1} + \cdots + \beta_{\delta_d}(t^{-1})^{-\delta_d}) \\
&= \left(\sum_{x=0}^{\infty}(-1)^x l_i^x (t^{-1})^x + \sum_{x=1}^{\infty}(-1)^{x-1} l_i^{x-1} l_j (t^{-1})^x\right)(\beta_0 + \beta_1(t^{-1})^{-1} + \cdots + \beta_{\delta_d}(t^{-1})^{-\delta_d}) \\
&= \left(\sum_{x=0}^{\infty} \sigma'_x (t^{-1})^x\right)(\beta_0 + \beta_1(t^{-1})^{-1} + \beta_2(t^{-1})^{-2} + \cdots + \beta_{\delta_d}(t^{-1})^{-\delta_d})
\end{aligned} \tag{19}$$

where $\sigma'_0 = 1$ and $\sigma'_x = (-1)^x l_i^x + (-1)^{x-1} l_i^{x-1} l_j$; it follows that $\sigma'_{x+1} = (-l_i)\sigma'_x$. Hence when we compute the coefficients γ_k of Q_d we see that the formula is identical to the case of type 1 generators. The difference between type 1 and type 2 generators when $n = d$ is that the minimal degree of the expression in Equation (19) is the same as the minimal degree of R_d. This is because the expression for $(t+l_i)^{-1}(t+l_j)$ written in terms of t^{-1}

has a constant term, which is not the case when $(t + l_i)^{-1}$ is written in terms of t^{-1}. Explicitly, we compute that

$$\gamma_k = \sum_{v=0}^{\delta_d - k} \sigma'_v \beta_{v+k}$$

which is identical to the expression in Equation (12) for type 1 generators except that the index begins at 0. Computations then yield

$$\gamma_k = -l_i \gamma_{k+1} + \sigma'_0 \beta_k$$

and hence

$$\gamma_{k+1} = -l_i^{-1}(\gamma_k - \beta_k).$$

We now state the following lemma, whose proof for $n \neq i$ is contained in the verification of the above expressions. The case $n = i$ is verified in Section 7.

Lemma 9. *Fix a type 2 generator s, and let $g \in \Gamma_d(q)$. Using the decompositions above in Equations (7) and (8), we see that:*

1. *If $n \neq i, j$ then the minimal degree of R_n is the same as the minimal degree of Q_n.*
2. *If $n = i$ then the minimal degree of Q_i is one less than the minimal degree of R_i.*
3. *If $n = j$ then the minimal degree of Q_j is one greater than the minimal degree of R_j.*

6.3. Construction of automata when $n \neq i$

We now construct, for each $1 \leq n \leq d,\ n \neq i$, and $\epsilon \in \{1, 2\}$ a finite state machine $\overline{\mathrm{M}_{n,\epsilon}}$ which accepts the convolution $\otimes(\sigma(R_n), \sigma(Q_n))$. The value of ϵ indicates whether s is a type 1 or type 2 generator. That is, the machine accepts strings of the form

$$\begin{pmatrix}\beta_1\\ \gamma_1\end{pmatrix}\begin{pmatrix}\beta_2\\ \gamma_2\end{pmatrix}\begin{pmatrix}\beta_3\\ \gamma_3\end{pmatrix}\cdots\begin{pmatrix}\beta_{\delta_n - 1}\\ \gamma_{\delta_n - 1}\end{pmatrix}\begin{pmatrix}\beta_{\delta_n}\\ \gamma_{\delta_n}\end{pmatrix} \tag{20}$$

except when ($\epsilon = 1$ and $n = d$) or ($\epsilon = 2$ and $n = j$) and in those two cases, strings of the form

$$\begin{pmatrix}\beta_1\\ \gamma_1\end{pmatrix}\begin{pmatrix}\beta_2\\ \gamma_2\end{pmatrix}\begin{pmatrix}\beta_3\\ \gamma_3\end{pmatrix}\cdots\begin{pmatrix}\beta_{\delta_n - 1}\\ \gamma_{\delta_n - 1}\end{pmatrix}\begin{pmatrix}\beta_{\delta_n}\\ \diamond\end{pmatrix} \tag{21}$$

where the relationship between the β_x and γ_x is explicitly described in the previous two sections. If $R_n = 0$ it follows that $Q_n = 0$ as well, and we adapt the machines to accept this pair as well.

For each $n \neq i$ construct a finite state machine $\overline{\mathrm{M}_{n,\epsilon}}$ as follows.

1. Create states $T_{\sigma,\tau}$ for all $\sigma, \tau \in \mathbb{Z}_q$, and for a given pair (σ, τ):
   (a) When $n \neq d$ and
      i. $\epsilon = 1$, compute the least residue γ of $-C_n^{-1}\tau + \sigma \pmod q$.
      ii. $\epsilon = 2$ and $n \neq j$, compute the least residue γ of $-C_n^{-1}\tau + D_n\sigma \pmod q$.

      For each $\beta \in \mathbb{Z}_q$, add a transition arrow from $T_{\sigma,\tau}$ with label $\binom{\beta}{\gamma}$ to state $T_{\beta,\gamma}$.
   (b) When $\epsilon = 2$ and $n = j$, for each $\beta \in \mathbb{Z}_q$, compute the least residue γ of $-C_n^{-1}\tau + \beta \pmod q$ and add a transition arrow from $T_{\sigma,\tau}$ with label $\binom{\beta}{\gamma}$ to state $T_{\beta,\gamma}$.
   (c) When $\epsilon = 1$, and $n = d$, compute the least residue γ of $-l_i^{-1}(\tau - \beta) \pmod q$ for each $\beta \in \mathbb{Z}_q$. Add a transition arrow from $T_{\sigma,\tau}$ to $T_{\beta,\gamma}$ with label $\binom{\beta}{\gamma}$.
   (d) When $\epsilon = 2$ and $n = d$, compute the least residue γ of $-l_i^{-1}(\tau - \sigma) \pmod q$. For each $\beta \in \mathbb{Z}_q$, add a transition arrow from $T_{\sigma,\tau}$ with label $\binom{\beta}{\gamma}$ to state $T_{\beta,\gamma}$.
2. Add a start state, and for each $\beta, \gamma \in \mathbb{Z}_q$, a transition edge with label $\binom{\beta}{\gamma}$ which terminates at $T_{\beta,\gamma}$. Allow the start state to be an accept state so that the empty pair $R_n = Q_n = 0$ is accepted.
3. Introduce an accept state A; from each state $T_{\sigma,\tau}$:
   (a) When $n \neq d$ and
      i. $\epsilon = 1$, compute the least residue γ of $-C_n^{-1}\tau + \sigma \pmod q$ and the least residue β of $C_n^{-1}\gamma \pmod q$.
      ii. $\epsilon = 2$ and $n \neq j$, compute the least residue γ of $-C_n^{-1}\tau + D_n\sigma \pmod q$ and the least residue β of $(C_nD_n)^{-1}\gamma \pmod q$.

      Add a single transition from this state to A with label $\binom{\beta}{\gamma}$.
   (b) When $\epsilon = 2$ and $n = j$, compute the least residue β of $-C_j^{-1}\tau$, and add a single transition to state A with label $\binom{\beta}{\diamond}$.
   (c) When $n = d$, and
      i. $\epsilon = 1$, let $\beta = \tau$ and add a single transition to state A with label $\binom{\beta}{\diamond}$.
      ii. $\epsilon = 2$, compute the least residue γ of $-l_i^{-1}(\tau - \sigma) \pmod q$. Add a single transition to state A with label $\binom{\gamma}{\gamma}$.

A word accepted by $\overline{\mathrm{M}_{n,\epsilon}}$ corresponds to two nonempty strings $\beta_1\beta_2\cdots\beta_k$ and $\gamma_1\gamma_2\cdots\gamma_\eta$ (for $\eta = k$ or $k-1$) where the coefficients differ according to Equation (10) or (18). Thus $\overline{\mathrm{M}_{n,\epsilon}}$ accepts the language of convolutions of the form $\otimes(\sigma(R_n), \sigma(Q_n))$, $n \neq i$ where R_n and Q_n arise from g and gs, respectively.

We now construct a machine M_ϵ which accepts $\otimes(\sigma(g), \sigma(h))$ if for each $1 \leq n \leq d$, $n \neq i$, the n-th substrings in the convolution $\otimes(\sigma(g), \sigma(h))$ are related in the manner proscribed by $\overline{\mathrm{M}_{n,\epsilon}}$. Recall from Lemmas 8 and 9 that for

- $n \neq i, j, d$, we have $|\sigma(R_n)| = |\sigma(Q_n)|$,
- $n = i$, we have $|\sigma(R_i)| + 1 = |\sigma(Q_i)|$,
- $n = j$, if $\epsilon = 1$ then $|\sigma(R_n)| = |\sigma(Q_n)|$ and if $\epsilon = 2$ then $|\sigma(R_d)| - 1 = |\sigma(Q_d)|$, and
- $n = d$, if $\epsilon = 1$ then $|\sigma(R_d)| - 1 = |\sigma(Q_d)|$, and if $\epsilon = 2$ then $|\sigma(R_d)| = |\sigma(Q_d)|$.

This list describes the offset in $\otimes(\sigma(g), \sigma(gs))$ between the strings $\sigma(R_n)$ on the top line of the convolution and $\sigma(Q_n)$ on the bottom line of the convolution when a finite state machine reads $\otimes(\sigma(g), \sigma(gs))$. If $\epsilon = 1$ then for $n > i$ the strings $\sigma(R_n)$ and $\sigma(Q_n)$ are offset by 1 as the convolution $\otimes(\sigma(g), \sigma(gs))$ is read. If $\epsilon = 2$ then for $i < k \leq j$ the strings $\sigma(R_k)$ and $\sigma(Q_k)$ are offset by 1, and for $k > j$ they are aligned, and hence read simultaneously.

Let Φ_n and Ψ_n be (possibly empty) strings of length $\eta_n \geq 0$ and $\chi_n \geq 0$ of symbols from the finite alphabet consisting of elements of $\mathbb{Z}_q$, for $1 \leq n \leq d$. Let $\mathcal{K}_\epsilon$ be the language of convolutions of the form

$$\otimes(\Phi_1 \# \Phi_2 \# \cdots \# \Phi_d, \Psi_1 \# \Psi_2 \# \cdots \# \Psi_d)$$

where for $n \neq i$ we assume without loss of generality (as these conditions can be verified with finite state automata) that the lengths η_k and χ_k agree with Lemma 8 if $\epsilon = 1$ and Lemma 9 if $\epsilon = 2$. We view the two strings in the convolution as arising from $\sigma(g)$ and $\sigma(h)$ for some $g, h \in \Gamma_d(q)$. It is clear that $\mathcal{K}_\epsilon$ is a regular language. We want to show that the subset $\mathcal{K}'_\epsilon$ of $\mathcal{K}_\epsilon$ in which $\otimes(\Phi_n, \Psi_n)$ is accepted by $\overline{\mathrm{M}_{n,\epsilon}}$, for all $1 \leq n \leq d$, $n \neq i$ is also a regular language.

Let $\overline{\mathcal{K}_\epsilon}$ consist of strings of the same form as those in $\mathcal{K}_\epsilon$ with the condition that $\eta_i = \chi_i$, that is, the strings Φ_i and Ψ_i have the same length. We impose no other conditions on the remaining strings Φ_n and Ψ_n. When $\epsilon = 1$, this language is accepted by the machine $\mathcal{M}_1$ constructed as follows.

1. The start state of $\mathcal{M}_1$ is the start state of the machine $\overline{\mathrm{M}_{1,1}}$.
2. For $1 \leq n \leq i-2$, add a transition arrow with label $\binom{\#}{\#}$ between the accept state A of $\overline{\mathrm{M}_{n,1}}$ and the start state of $\overline{\mathrm{M}_{n+1,1}}$.
3. Add a transition arrow with label $\binom{\#}{\#}$ from the start state of $\overline{\mathrm{M}_{n-1,1}}$ to the start state of $\overline{\mathrm{M}_{n,1}}$, for all $2 \leq n \leq d$ with $n \neq i$.
4. Add a transition arrow with label $\binom{\#}{\#}$ from the accept state A of $\overline{\mathrm{M}_{i-1,1}}$ to a state S_i. Add a loop at S_i with label $\binom{\beta}{\gamma}$ for each pair $\beta, \gamma \in \mathbb{Z}_q$. From S_i add a transition arrow to the start state of $\overline{\mathrm{M}_{i+1,1}}$ with label $\binom{\#}{\#}$.
5. For $i+1 \leq n \leq d-1$, add a transition arrow with label $\binom{\#}{\#}$ between the accept state A of $\overline{\mathrm{M}_{n,1}}$ and the start state of $\overline{\mathrm{M}_{n+1,1}}$.
6. Let the accept state A of $\overline{\mathrm{M}_{d,1}}$ be the accept state of the entire machine.

It then follows from Lemma 4 that $\mathcal{K}'_1$ is a regular language. This is exactly the language of convolutions $\otimes(\sigma(g), \sigma(h))$ where all but the i-th substrings have the same relationship as if $h = gs$ where s is a type 1 generator.

When $\epsilon = 2$ we assume that in $\overline{\mathcal{K}_2}$ the strings Φ_i and Ψ_i have the same length, as do the strings Φ_j and Ψ_j. We construct M_2 as above, with one modification. Introduce a state S_j which replaces $\overline{\mathrm{M}_{j,2}}$ analogous to S_i above. The resulting machine accepts convolutions where all but the i-th and j-th substrings differ in the proscribed manner for a type 2 generator. The language accepted by this machine is regular, and it follows from Lemma 4 that $\mathcal{K}_2'$, in which $|\Psi_i| = |\Phi_i|+1$ and $|\Psi_j| = |\Phi_j|-1$, is a regular language. Create a simple finite state machine which additionally verifies that Φ_j and Ψ_j have the relationship given by $\overline{\mathrm{M}_{j,2}}$. We then conclude that the language of convolutions $\mathcal{K}_2$ of the form $\otimes(\sigma(g), \sigma(h))$ where all but the i-th substrings have the same relationship as if $h = gs$ where s is a type 2 generator, is a regular language.

7. Construction of automata II

We now determine the relationship between the coefficients of R_i and Q_i arising from $\sigma(g)$ and $\sigma(gs)$, where i is fixed by the choice of generator s. The coefficient of $(t+l_i)^{-1}$ in Q_i is given by a complicated sum, and we construct a separate automaton to verify that this coefficient is correct. A second automaton is constructed to verify the relationship between the remaining coefficients of R_i and Q_i.

7.1. Analysis of coefficients for type 1 generators

Suppose that $\sigma(g)$ and $\sigma(gs)$ are as above, where s is a type 1 generator. First we compute the coefficient of $(t+l_i)^{-1}$ in Q_i, beginning with the expression in Equation (8). We compute this coefficient as a running sum, which we refer to as a partial sum σ, as each term in each R_n for $n \neq i$ will contribute a term to the final sum which becomes the coefficient of $(t+l_i)^{-1}$ in Q_i.

First note that the $b(t+l_i)^{-1}$ term in Equation (8) will contribute b to σ. Additionally, $(t+l_i)^{-1}R_i$ only contains terms of degree at most -2 and hence does not contribute any terms to σ.

We now compute the contribution to σ from a generic term in $(t+l_i)^{-1}R_n$ for $n \neq i$; when $n \neq d$ as well, such a term has the form $(t+l_i)^{-1}\xi(t+l_n)^{-e}$ where $\xi \in \mathbb{Z}_q$ and $e \in \mathbb{N}$. Using Equation (2) or Equation (4) (when $n = d$) we see that

$$(t+l_i)^{-1}\xi(t+l_n)^{-e} = (t+l_i)^{-1}\xi\sum_{r=0}^{\infty}\chi_r(t+l_i)^r$$

which has a single term of negative degree, namely $\xi\chi_0(t+l_i)^{-1}$. Here, χ_r is the coefficient computed in Equation (2) and $\chi_0 = C_n^e$. When $n = d$ we obtain an analogous equation with χ_0 computed as in Equation (4), in which case $\chi_0 = (-l_i)^e$. So each term of R_n contributes its constant term (when expanded in the variable $t+l_i$) to the sum σ. As above, write

$$\begin{aligned} R_n &= \sum_{z=1}^{\delta_n}\beta_{n,z}(t+l_n)^{-z} \\ &= \sum_{z=1}^{\delta_n}\beta_{n,z}\sum_{r=0}^{\infty}\binom{z}{r}C_n^{r-z}(t+l_i)^r \end{aligned}$$

when $n \neq d$, and when $n = d$,

$$R_d = \sum_{z=1}^{\delta_d} \beta_{d,z} \sum_{r=0}^{z} \binom{z}{r} (-l_i)^{z-r}(t+l_i)^r.$$

As we are only interested in the terms in the above sum when $r = 0$, we see that the coefficient of $(t + l_i)^{-1}$ in Q_i must be

$$b + \sum_{v=1, v\neq i}^{d-1} \sum_{z=1}^{\delta_v} \beta_{v,z} C_t^z + \sum_{w=0}^{\delta_d} \beta_{d,w}(-l_i)^w \tag{22}$$

where the values of b and i are determined by the original generator s.

Next, we ignore the coefficient of $(t + l_i)^{-1}$ and then the above reasoning shows that the terms of Q_i of degree at most -2 are given by the terms of $\mathcal{L}S_i((t+l_i)^{-1}R_i)$. Write

$$(t+l_i)^{-1}R_i = (t+l_i)^{-1} \sum_{z=1}^{\delta_i} \beta_{i,z}(t+l_i)^{-z} = \sum_{z=1}^{\delta_i} \beta_{i,z}(t+l_i)^{-(z+1)}$$

and notice that the coefficients of R_i are shifted over to form the coefficients of Q_i of degree at most -2. The minimal degree of Q_i is $-(\delta_i + 1)$ with coefficient $\beta_{i,\delta_i} \neq 0$. Thus we are comparing strings of coefficients of the form

$$\begin{matrix} \beta_{i,1} & \beta_{i,2} & \beta_{i,3} & \cdots & \beta_{i,\delta_i-1} & \beta_{i,\delta_i} & \# \\ \xi & \beta_{i,1} & \beta_{i,2} & \cdots & \beta_{i,\delta_i-2} & \beta_{i,\delta_i-1} & \beta_{i,\delta_i} \end{matrix} \tag{23}$$

where $\xi \in \mathbb{Z}_q$ can be any element, since we are not concerned in this step with the relationship between the coefficient of $(t + l_i)^{-1}$ in R_i and Q_i.

7.2. Construction of finite state machines for type 1 generators when $n = i$

Consider again the language $\mathcal{K}_1$ defined in Section 6; this is the language of all possible strings $\otimes(\sigma(g), \sigma(h))$ for $g, h \in \Gamma_d(q)$ and s a type 1 generator. These strings have the form $\Phi_1\#\Phi_2\#\cdots\#\Phi_d$ and $\Psi_1\#\Psi_2\#\cdots\#\Psi_d$, respectively, where each Φ_k and Ψ_k is a string of elements of $\mathbb{Z}_q$, and we assume without loss of generality that the lengths of the corresponding substrings are related as in Lemma 8.

In this section, we must show that the subset $\mathcal{H}_1$ of $\mathcal{K}_1$ in which the first entry of Ψ_i relates to the entire string $\Phi_1\#\Phi_2\#\cdots\#\Phi_d$ as specified in Equation (22), and then the subset $\mathcal{H}_2$ of strings where the remaining entries of Φ_i and Ψ_i differ as in Equation (23), are regular languages. Their intersection is then a regular language $\mathcal{H}$ in which Φ_i and Ψ_i differ as in $\otimes(\sigma(g), \sigma(gs))$ where s is a type 1 generator.

We first construct a machine $\mathrm{M}_{\mathcal{H},1}$ which accepts exactly the set $\mathcal{H}_1$. This machine stores a partial sum which is augmented as each pair $\binom{\beta}{\gamma}$ is read from $\otimes(\sigma(g), \sigma(h)) \in \mathcal{K}_1$, although only the value of β increases the sum. To accept a string, this value is compared against the first entry in $\Psi_i \subset \sigma(h)$. As the machine has no memory, these values are stored implicitly in the indexing of the states and the transition functions.

For each value of n with $1 \leq n \leq d$, $n \neq i$, consider the cycle $\mathcal{C}_n = \{C_n, C_n^2, C_n^3 \cdots, C_n^{k_n} = 1\}$ of length k_n where all values are taken modulo q, and C_n is defined in Section 4. Create a set of $q^2 k_n$ states of the form $T_{\alpha,e,\sigma}$ where $\alpha, \sigma \in \mathbb{Z}_q$, $e \in \{1, 2, 3, \cdots, k_n\}$, where

- α stores the coefficient of the term of R_n that we are reading (denoted $\beta_{n,z}$ above),
- e is the exponent of C_n in the cycle $\mathcal{C}_n$, and
- σ is the partial sum of the coefficient of $(t + l_i)^{-1}$ in Q_i.

From state $T_{\alpha,e,\sigma}$, for each $\beta \in \mathbb{Z}_q$ and $\gamma \in \mathbb{Z}_q \cup \{\#\}$, compute σ' to be the least residue of $\sigma + \beta C_n^{e+1} \pmod q$ (resp. $\sigma + \beta(-l_i)^{e+1} \pmod q$ when $n = d$) and introduce a transition labeled $\binom{\beta}{\gamma}$ to $T_{\beta,e+1,\sigma'}$. To streamline notation, we always assume that the second coordinate in the state index is reduced modulo k_n, and the third index is reduced modulo q. Denote the resulting machine N_n; note that this step creates $d - 1$ distinct finite state automata, and that we have not added any start or accept states to this machine yet.

To create the composite machine $\mathrm{M}_{\mathcal{H},1}$, begin with a start state with q^2 transition arrows emanating from it, with labels $\binom{\beta}{\gamma}$ for all $\beta, \gamma \in \mathbb{Z}_q$. The arrow with label $\binom{\beta}{\gamma}$ terminates at state $T_{\beta,1,b+\beta C_1}$ in N_1.

To transition from N_c to N_{c+1} for $1 \leq c \leq i - 2$, create a set of q states $S_{c,0}, S_{c,1}, S_{c,2}, \cdots S_{c,q-1}$ reflecting in the second coordinate the possible values of the partial sum σ. From each state $T_{\alpha,e,\sigma}$ of N_c introduce a transition with label $\binom{\#}{\#}$ to the state $S_{c,\sigma}$. From each state $S_{c,\sigma}$ introduce q^2 transitions, where the transition with label $\binom{\beta}{\gamma}$ terminates at the state $S_{\beta,1,\sigma+\beta C_{c+1}^1}$ of N_{c+1}, for each pair $\beta, \gamma \in \mathbb{Z}_q$. As we pass from N_c to N_{c+1} in this way we are transitioning from reading the coefficients of R_c to the coefficients of R_{c+1} and the information we must retain in terms of the state indexing is the partial sum σ.

Now add one additional arrow from the start state with label $\binom{\#}{\#}$ which terminates at state $S_{1,b}$ where b is fixed in the generator s. From each state $S_{c,p}$ add a transition with label $\binom{\#}{\#}$ which terminates at state $S_{c+1,p}$. This corresponds to $\Phi_c = \emptyset$ for $c \leq i - 2$.

Now we have connected the machines N_1 through N_{i-1}. As above, create q states labeled $S_{i-1,0}, S_{i-1,1}, S_{i-1,2}, \cdots S_{i-1,q-1}$ reflecting the possible values of σ in the second coordinate. From each state $T_{\alpha,e,\sigma}$ of N_{i-1} introduce a transition with label $\binom{\#}{\#}$ to the state $S_{i-1,\sigma}$. Create q^2 states $S_{i,a,b}$ for each pair $\alpha, \psi \in \mathbb{Z}_q$; the index α stores the value of σ and the index ψ is the first entry $\psi_{i,1}$ in Ψ_i, which corresponds to the coefficient of $(t + l_i)^{-1}$ in Q_i. From state $S_{i-1,c}$ for each pair $\alpha, \psi \in \mathbb{Z}_q$ add a transition with label $\binom{\alpha}{\psi}$ to state $S_{i,c,\psi}$. At each state $S_{i,a,b}$ add a loop with label $\binom{a}{b}$ for $a \in \mathbb{Z}_q \cup \{\#\}$ and $b \in \mathbb{Z}_q$.

Next, create q copies of N_{i+1}, which we denote $\mathrm{N}_{i+1,p}$ for $p \in \mathbb{Z}_q$. No transitions will be introduced between copies of $\mathrm{N}_{i+1,p}$ and $\mathrm{N}_{i+1,p'}$ for $p \neq p'$. From state $S_{i,\sigma,\psi}$, add a transition arrow with label $\binom{\beta}{\#}$ to the state $T_{\beta,1,\sigma+\beta C_i}$ of $\mathrm{N}_{i+1,\psi}$. The lack of transitions between the $\mathrm{N}_{i,p}$ retains the value of ψ.

Create q copies of N_r for $i+2 \leq r \leq d$ which we index by $\mathrm{N}_{r,p}$ for $p \in \mathbb{Z}_q$, and corresponding transition states $S_{r,p}$ as above. Mimic the transitions between machines as above, with two changes. To connect the machines $\mathrm{N}_{r,p}$ and $\mathrm{N}_{r+1,p}$ via the intermediate states $S_{k,p}$,

- replace any transition with label $\binom{\#}{\#}$ from $\mathrm{N}_{r,p}$ to $S_{k,p}$ by a set of transitions with labels $\binom{\#}{\xi}$, for $\xi \in \mathbb{Z}_q$, and
- replace any transition with label $\binom{a}{b}$ with $a, b \in \mathbb{Z}_q$ from $S_{k,p}$ to $\mathrm{N}_{r+1,p}$ by a set of transitions with labels $\binom{\chi}{\#}$, for $\chi \in \mathbb{Z}_q$.

To finish the construction of the machine $\mathrm{M}_{\mathcal{H},1}$ which accepts the language $\mathcal{H}_1$, we must verify that the final sum σ is exactly the coefficient $\psi_{i,1}$. To accomplish this, in $\mathrm{N}_{d,p}$ designate only $T_{\alpha,e,p}$ as an accept state.

It follows directly from Lemma 5 that the language of convolutions of strings $\otimes(\sigma(g), \sigma(h))$ for which all but the initial coefficients in Φ_i and Ψ_i are related as in Equation (23) form a regular language $\mathcal{H}_2$. Let $\mathcal{H} = \mathcal{H}_1 \cap \mathcal{H}_2$; then $\mathcal{H}$ is the language of those convolutions where Φ_i and Ψ_i are related as in $\otimes(\sigma(g), \sigma(gs))$ where s is a type 1 generator. Hence $\mathcal{N}'_s = \mathcal{K}_1 \cap \mathcal{H}$ is a regular language which contains all convolutions of the form $\otimes(\sigma(g), \sigma(gs))$.

7.3. Analysis of coefficients for type 2 generators

We now mimic the analysis of the coefficients of $\sigma(R_i)$ and $\sigma(Q_i)$ where R_i and Q_i arise from $\sigma(g)$ and $\sigma(gs)$ and s is a type 2 generator. Recall that we must convert

$$b(t+l_i)^{-1} + (t+l_i)^{-1}(t+l_j)(R_1 + R_2 + \cdots + R_d)$$

to a Laurent polynomial in the variable $t+l_i$. As before, the coefficient of $(t+l_i)^{-1}$ in Q_i will be computed as a running sum σ, of which b will be a summand.

Note that $(t+l_i)^{-1}(t+l_j) = 1 + (l_j - l_i)(t+l_i)^{-1}$ and the initial 1 creates the difference between the case when s is a type 1 generator and this case. In particular, when we compute $(t+l_i)^{-1}(t+l_j)R_i$ we see that

$$\left(1 + (l_j - l_i)(t+l_i)^{-1}\right) \sum_{z=1}^{\delta_i} \beta_{i,z}(t+l_i)^{-z} = \sum_{z=1}^{\delta_i+1} \tau_z (t+l_i)^{-z} \tag{24}$$

where

- $\tau_1 = \beta_{i,1}$,
- $\tau_z = \beta_{i,z} + (l_j - l_i)\beta_{i,z-1}$ for $2 \leq z \leq \delta_i$, and
- $\tau_{\delta_i+1} = (l_j - l_i)\beta_{i,\delta_i}$.

We notice immediately that this Laurent polynomial contributes its initial coefficient, $\beta_{i,1}$ to the coefficient σ of $(t+l_i)^{-1}$ in Q_i, which is not the case when s is a type 1 generator.

We now compute the contribution to the sum σ from a generic term in the expression $(1+(l_j-l_i)(t+l_i)^{-1})R_n$ for $n \neq i, d$, which is of the form

$$(1+(l_j-l_i)(t+l_i)^{-1})\xi(t+l_n)^{-e}$$

for $\xi \in \mathbb{Z}_q$. Using Equation (2) we see that

$$\begin{aligned}
&\left(1+(l_j-l_i)(t+l_i)^{-1}\right)\xi(t+l_n)^{-e}\\
&=(1+(l_j-l_i)(t+l_i)^{-1})\xi\sum_{r=0}^{\infty}\binom{-e}{r}(l_i-l_n)^{-e-r}(t+l_i)^r\\
&=(1+(l_j-l_i)(t+l_i)^{-1})\xi\sum_{r=0}^{\infty}\chi_r(t+l_i)^r
\end{aligned}$$

in which the coefficient of $(t+l_i)^{-1}$ is $(l_j-l_i)\xi\chi_0=(l_j-l_i)\xi C_n^e$. As this differs from the case when s is a type 1 generator only by a constant, namely l_j-l_i, we see that the identical analysis applies to computing the contribution to σ from $(1+(l_j-l_i)(t+l_i)^{-1})R_n$ when $n \neq i, d$. When $n=d$, we use Equation (4) instead of Equation (2) to obtain

$$(1+(l_j-l_i)(t+l_i)^{-1})\xi(t^{-1})^{-e}=(1+(l_j-l_i)(t+l_i)^{-1})\xi\sum_{r=0}^{e}\binom{e}{r}(-l_i)^{e-r}(t+l_i)^n.$$

From this we see that each term of this form contributes $(l_i-l_j)\xi(-l_i)^e$ to the sum σ, and the constant term contributes $(l_j-l_i)\beta_{d,0}$ to this sum.

Thus the coefficient of $(t+l_i)^{-1}$ in Q_i is

$$b+\beta_{i,1}+(l_j-l_i)\left(\sum_{v=1,v\neq i}^{d-1}\sum_{z=1}^{\delta_v}\beta_{v,z}C_v^z+\sum_{z=0}^{\delta_d}\beta_{d,z}(-l_i)^z\right) \tag{25}$$

where the values of i, j and b are determined by the original generator s. This expression is very close to the one in Equation (22): there are two initial terms instead of one, and the summation is multiplied by a constant. The automaton constructed in Section 7.2 is easily adapted to account for these minor changes in the sum, and we obtain the same conclusions as in Section 7.1 for type 2 generators.

It now follows that the terms of Q_i of degree at most -2 are given by the terms of

$$(t+l_i)^{-1}(t+l_j)R_i=(1+(l_j-l_i)(t+l_i)^{-1})R_i=\sum_{z=1}^{\delta_i+1}\tau_z(t+l_i)^{-z}$$

for τ_z defined in Equation (24). Letting $D_i = l_j - l_i$ we create a simple automaton which accepts strings of the form:

$$\binom{\beta_{i,1}}{\beta_{i,1}}\binom{\beta_{i,2}}{D_i\beta_{i,1}+\beta_{i,2}}\binom{\beta_{i,3}}{D_i\beta_{i,2}+\beta_{i,3}}\cdots\binom{\beta_{i,\delta_i}}{D_i\beta_{i,\delta_i-1}+\beta_{i,\delta_i}}\binom{\#}{D_i\beta_{i,\delta_i}} \tag{26}$$

where all coordinates are computed modulo q. Create q^2 states $T_{a,b}$ for each $a, b \in \mathbb{Z}_q$. From a start state S, add transition arrows with label $\binom{a}{a}$ terminating at state $T_{a,a}$. From state $T_{a,b}$, compute d to be the least residue mod q of $aD_i + c$, for each $c \in \mathbb{Z}_q$ and add a transition with label $\binom{c}{d}$ which terminates at state $T_{c,d}$. From each state $T_{a,b}$ add a transition with label $\binom{\#}{aD_i}$ to an accept state. Then this machine can be easily extended to a machine which verifies that in strings $\Phi = \Phi_1\#\Phi_2\#\cdots\#\Phi_d$ and $\Psi = \Psi_1\#\Psi_2\#\cdots\#\Psi_d$, all but the initial entries of Φ_i and Ψ_i differ as in expression (26).

Thus we have shown that the set of all convolutions $\otimes(\sigma(g), \sigma(h))$ for which the coefficients of $\sigma(g)$ and the initial coefficient of Ψ_i are related as in Equation (25) form a regular language, as do the set of convolutions $\otimes(\sigma(g), \sigma(h))$ for which all but the initial coefficients in Φ_i and Ψ_i are related as in Equation (26). Thus the intersection of these languages is a regular language $\mathcal{H}$. As when s was a type 1 generator, we conclude that $\mathcal{N}'_s = \mathcal{K}_2 \cap \mathcal{H}$ is a regular language which contains all convolutions of the form $\otimes(\sigma(g), \sigma(gs))$ where s.

Regardless of whether s is a type 1 or type 2 generator, to complete the proof of Theorem 6 we must show that if $\mathcal{N}_s = \mathcal{P}_s\mathcal{N}'_s$, then $\mathcal{L}_s = \mathcal{N}_s$, where $\mathcal{L}_s$ is the multiplier language for the generator s. It is clear that $\mathcal{L}_s \subseteq \mathcal{N}_s$. We now prove the reverse inclusion.

Let $\otimes(p_1, p_2) \in \mathcal{P}_s$ and

$$\otimes(\Phi_1\#\Phi_2\#\cdots\#\Phi_d, \Psi_1\#\Psi_2\#\cdots\#\Psi_d) \in \mathcal{N}'_s.$$

Suppose that g (resp. h) in $\Gamma_d(q)$ has $\pi(g) = p_1$ (resp. $\pi(h) = p_2$) and $\sigma(g) = \Phi_1\#\Phi_2\#\cdots\#\Phi_d$ (resp. $\sigma(h) = \Psi_1\#\Psi_2\#\cdots\#\Psi_d$). We will show that $h = gs$.

We can write g and h in matrix form, where the entries of p_1 and p_2, respectively, determine the exponents in the upper left entry of each matrix. Let $p_1 = (m_1, m_2, \cdots, m_{d-1})$ and $p_2 = (n_1, n_2, \cdots, n_{d-1})$. We know that these strings differ in the manner proscribed by $\mathcal{P}_s$ which corresponds to multiplication by s. The upper right entry of the matrix representing g is then $R = \Pi_{r=1}^{d-1}(t + l_r)^{m_r}(R_1 + R_2 + \cdots + R_d)$, where the coefficients of R_k are given by Φ_k. We analogously construct the upper right polynomial entry of the matrix for h as $Q = \Pi_{r=1}^{d-1}(t + l_n)^{n_r}(Q_1 + Q_2 + \cdots + Q_d)$, where the coefficients of Q_k are given by Ψ_k.

Now consider the matrix for the element gs. Since $\otimes(p_1, p_2) \in \mathcal{P}_s$, we must have $\pi(gs) = (n_1, n_2, \cdots, n_d) = \pi(h)$. If the polynomial in the upper right entry of gs is denoted P, use the Decomposition Lemma to write

$$\Pi_{k=1}^{d-1}(t + l_k)^{-n_k}P = P_1 + P_2 + \cdots + P_d.$$

In order to show that $h = gs$, we must verify that the polynomial entries are identical. First note that the values of the minimal degrees of R_k, Q_k and P_k are encoded in the lengths of the substrings of $\sigma(g), \sigma(h)$ and $\sigma(gs)$ respectively. As both $\otimes(\sigma(g), \sigma(h))$ and $\otimes(\sigma(g), \sigma(gs))$ are accepted strings in $\mathcal{N}'_s$, the minimal degrees of Q_k and P_k are identical for $1 \le k \le d$.

Suppose that s is a type 1 generator, and $n \neq i, d$. We will show that $Q_n = P_n$. From Table 1 we see that the value of β_{δ_n} in R_n determines the coefficient of the minimal degree term in both Q_n and P_n, since both $\otimes(g, h)$ and $\otimes(g, gs)$ are accepted strings in $\mathcal{N}'_s$. Hence the coefficient of minimal degree in these two polynomials is identical. The following equations determine the coefficients of the polynomials P_n for $n \neq i$ in increasing order of degree, namely

$$\gamma_k = \begin{cases} -C_n\gamma_{k+1} + C_n\beta_k & \text{if } n \neq i, d \\ -l_i\gamma_{k+1} + \alpha'_1\beta_{k+1} & \text{if } n = d \end{cases}$$

and hence the remaining coefficients of both Q_n and P_n are determined by the coefficients of R_n and the coefficients of higher degree in each polynomial, which are identical. Thus these two polynomials are identical. This reasoning can be adapted both to the case $n = d$ and to type 2 generators using the following formulae from Section 6.2.

$$\gamma_k = \begin{cases} -C_n\gamma_{k+1} + C_nD_n\beta_k & \text{if } n \neq i, j, d \\ -C_j\gamma_{k+1} - \beta_{k+1} & \text{if } n = j \\ -l_i\gamma_{k+1} + \sigma'_0\beta_k & \text{if } n = d \end{cases}$$

It remains to verify that $Q_i = P_i$. When s is a type 1 generator, the coefficients of the terms of degree less than -1 in both Q_i and P_i are simply a translate of the coefficients of R_i, hence identical. When s is a type 2 generator, the coefficients of the terms of degree less than -1 in both Q_i and P_i are uniquely determined by the coefficients of R_i, and hence identical. Regardless of the type of generator, Equations (22) and (25) demonstrate that the initial coefficient of P_i and Q_i only depends on the entries of $\sigma(g)$ and hence must be identical. Hence $h = gs$ and it follows that $\mathcal{N}_s = \mathcal{L}_s$. This finishes the proof of Theorem 6 that the multiplier languages are regular for each generator $s \in S_{d,q}$, and we conclude that $\Gamma_d(q)$ is graph automatic.

References

[1] Laurent Bartholdi, Markus Neuhauser, Wolfgang Woess, Horocyclic products of trees, J. Eur. Math. Soc. (JEMS) 10 (3) (2008) 771–816.

[2] Dmitry Berdinsky, Bakhadyr Khoussainov, On Automatic Transitive Graphs, Springer International Publishing, Cham, 2014, pp. 1–12.

[3] Yves de Cornulier, Romain Tessera, Metabelian groups with quadratic Dehn function and Baumslag–Solitar groups, Confluentes Math. 2 (4) (2010) 431–443.

[4] Reinhard Diestel, Imre Leader, A conjecture concerning a limit of non-Cayley graphs, J. Algebraic Combin. 14 (1) (2001) 17–25.

[5] Murray Elder, Jennifer Taback, $\mathcal{C}$-graph automatic groups, J. Algebra 413 (2014) 289–319.

[6] Murray Elder, Jennifer Taback, Thompson's group F is 1-counter graph automatic, Groups Complex. Cryptol. 8 (1) (2016) 21–33.

[7] David B.A. Epstein, James W. Cannon, Derek F. Holt, Silvio V.F. Levy, Michael S. Paterson, William P. Thurston, Word Processing in Groups, Jones and Bartlett Publishers, Boston, MA, 1992.

[8] Alex Eskin, David Fisher, Kevin Whyte, Coarse differentiation of quasi-isometries I: spaces not quasi-isometric to Cayley graphs, Ann. of Math. (2) 176 (1) (2012) 221–260.

[9] Benson Farb, Automatic groups: a guided tour, Enseign. Math. (2) 38 (3–4) (1992) 291–313.

[10] Mikhail Gromov, Asymptotic invariants of infinite groups, in: Geometric Group Theory, Vol. 2, Sussex, 1991, in: London Math. Soc. Lecture Note Ser., vol. 182, Cambridge Univ. Press, Cambridge, 1993, pp. 1–295.

[11] Derek F. Holt, Automatic groups, subgroups and cosets, in: The Epstein Birthday Schrift, in: Geom. Topol. Monogr., vol. 1, Geom. Topol. Publ., Coventry, 1998, pp. 249–260 (electronic).

[12] Martin Kassabov, Timothy R. Riley, The Dehn function of Baumslag's metabelian group, Geom. Dedicata 158 (2012) 109–119.

[13] Olga Kharlampovich, Bakhadyr Khoussainov, Alexei Miasnikov, From automatic structures to automatic groups, Groups Geom. Dyn. 8 (1) (2014) 157–198.

[14] Walter Neumann, Michael Shapiro, A short course in geometric group theory. Topology Atlas, iaai-13, January 1996.

[15] Melanie Stein, Jennifer Taback, Peter Wong, Automorphisms of higher rank lamplighter groups, arXiv:1412.2271 [math.GR], 2014 Dec.

[16] Melanie Stein, Jennifer Taback, Metric properties of Diestel–Leader groups, Michigan Math. J. 62 (2) (2013) 365–386.

[17] Melanie Stein, Jennifer Taback, Peter Wong, Automorphisms of higher rank lamplighter groups, Internat. J. Algebra Comput. 25 (8) (2015) 1275–1299.

[18] Jennifer Taback, Sharif Younes, Tree-based language complexity of Thompson's group F, Groups Complex. Cryptol. 7 (2) (2015) 135–152.