

Elsevier required licence: © <2021>. This manuscript version is made available under the CC-BY-NC-ND 4.0 license <http://creativecommons.org/licenses/by-nc-nd/4.0/>

The definitive publisher version is available online at

[\[https://www.sciencedirect.com/science/article/abs/pii/S0890540121000833?via%3Dihub\]](https://www.sciencedirect.com/science/article/abs/pii/S0890540121000833?via%3Dihub)

Cayley Polynomial–Time Computable Groups

Dmitry Berdinsky^{a,b}, Murray Elder^{c,1}, Prohrak Kruengthomya^{a,b}

^a*Department of Mathematics, Faculty of Science, Mahidol University, Bangkok, 10400, Thailand*

^b*Centre of Excellence in Mathematics, Commission on Higher Education, Bangkok, 10400, Thailand*

^c*School of Mathematical and Physical Sciences, University of Technology Sydney, Ultimo, NSW 2007, Australia*

Abstract

We propose a new generalisation of Cayley automatic groups, varying the time complexity of computing multiplication, and language complexity of the normal form representatives. We first consider groups which have normal form language in the class \mathcal{C} and multiplication by generators computable in linear time on a certain restricted Turing machine model (position–faithful one-tape). We show that many of the algorithmic properties of automatic groups are preserved (quadratic time word problem), prove various closure properties, and show that the class is quite large; for example it includes all virtually polycyclic groups. We then generalise to groups which have normal form language in the class \mathcal{C} and multiplication by generators computable in polynomial time on a (standard) Turing machine. Of particular interest is when $\mathcal{C} = \text{REG}$ (the class of regular languages). We prove that $\text{REGCayley polynomialtime computable groups}$ includes all finitely generated nilpotent groups, the wreath product $\mathbb{Z}_2 \wr \mathbb{Z}^2$, and Thompsons group F .

Keywords: Cayley position–faithful linear–time computable group; Cayley polynomialtime computable group; position–faithful one–tape Turing machine; Cayley distance function

Email addresses: berdinsky@gmail.com (Dmitry Berdinsky), murray.elder@uts.edu.au (Murray Elder), prohrakju@gmail.com (Prohrak Kruengthomya)

¹The second author is supported by Australian Research Council grant DP160100486.

1. Introduction

How one can represent elements of an infinite finitely generated group G ? A natural way to do this is to assign for each group element $g \in G$ a unique normal form which is a string over some finite alphabet (not necessarily a generating set). Kharlampovich, Khoushainov and Miasnikov used this approach to introduce the notion of a Cayley automatic group [1] that naturally extends the classical notion of an automatic group introduced by Thurston and others [2]. They require that the language of normal forms to be regular, and that for each s from some finite set of semigroup generators $S \subset G$ there is a two-tape synchronous automaton recognizing all pairs of strings (u, v) for which u is the normal form of some group element g and v is the normal form of the group element gs . Case, Jain, Seah and Stephan showed that this is equivalent to the existence of a position-faithful² one-tape Turing machine for each $s \in S$ which computes the output v from the input u in linear time [3]. Is it possible to extend the notion of a Cayley automatic group which admit normal forms satisfying this (linear time) property? Can it be extended further requiring not linear but polynomial time?

In this paper we consider groups which admit normal forms from some formal language class (not necessarily regular) where multiplication satisfies the (linear time) and (polynomial time) properties. We study their algorithmic and closure properties. We analyse examples of such groups and their normal forms. Furthermore, we study the characterization of these normal forms in terms of the *Cayley distance function* (as defined in [4] and studied in [5, 6]). In particular, we investigate examples of non-automatic groups for which this function can be diminished to the zero function for some normal forms satisfying either (linear time) or (polynomial time) properties. This is quite different to the situation for Cayley automatic representations of these groups, where the Cayley distance function is always separated from the zero function by some unbounded nondecreasing function which depends only on the group.

Contribution and paper outline. In Section 2 we recall the notion of a Cayley automatic group and a Cayley automatic representation.

In Section 3 we introduce the notion of a \mathcal{C} -Cayley position-faithful (p.f.) linear-time computable group and a \mathcal{C} -Cayley p.f. linear-time computable representation, for a given class of languages \mathcal{C} , where we require a language of

²see Definition 2

normal forms to be in the class \mathcal{C} and right multiplication by each semigroup generator to be computed by a position-faithful one-tape Turing machine in linear time.

We show that \mathcal{C} -Cayley p.f. linear-time computable groups preserve some key properties of Cayley automatic groups. In Theorem 1 we show that each \mathcal{C} -Cayley p.f. linear-time computable representation has quasigeodesic normal form. In Theorem 2 we show that there is a quadratic time algorithm computing this normal form. The latter implies that for every \mathcal{C} -Cayley p.f. linear-time computable group the word problem is decidable in quadratic time, see Corollary 1. In Theorems 4, 5 and 6 we prove that under some very mild restrictions on the class \mathcal{C} , the family of \mathcal{C} -Cayley p.f. linear-time computable groups is closed under taking finite extension, direct product and free product, respectively. Furthermore, in Theorem 8 we show that (under similar mild restrictions) the family of \mathcal{C} -Cayley p.f. linear-time computable groups is contained in the family of \mathcal{C} -graph automatic groups introduced by the second author and Taback. The collection of all \mathcal{C} -Cayley p.f. linear-time computable groups for all classes \mathcal{C} forms the family of *Cayley p.f. linear-time computable groups*. In Theorem 7 we show that the family of Cayley p.f. linear-time computable groups is closed under taking finitely generated (f.g.) subgroups. In Theorem 3 we notice that for each Cayley p.f. linear-time computable representation the language of normal forms must be recursively enumerable. Moreover, in Proposition 2 we give an example of a Cayley p.f. linear-time computable representation for which the language of normal forms is not recursive. In Theorem 9 we show that the family of Cayley p.f. linear-time computable groups comprises all f.g. subgroups of $\text{GL}(n, \mathbb{Q})$; in particular, it includes all polycyclic groups.

In Section 4 we consider further generalization of Cayley p.f. linear-time computable groups – the notion of a \mathcal{C} -Cayley polynomial-time computable group and a \mathcal{C} -Cayley polynomial-time computable representation, for a given class of languages \mathcal{C} , where we require a language of normal forms to be in the class \mathcal{C} and the right multiplication by each semigroup generator to be computed by a one-tape Turing machine in polynomial time. We note that a \mathcal{C} -Cayley polynomial-time computable representation does not necessary have quasigeodesic normal form (in contrast to the p.f. linear-time case). However, assuming that a \mathcal{C} -Cayley polynomial-time computable representation has quasigeodesic normal form, in Theorem 10 we show that there is a polynomial-time algorithm computing this normal form. The latter implies that the word problem is decidable in polynomial time, see Corollary

3. In Theorem 12 we notice that, similarly to \mathcal{C} -Cayley p.f. linear-time computable groups, the families of \mathcal{C} -Cayley polynomial-time computable groups and the ones with quasigeodesic normal forms are closed under taking a finite extension, direct product and free product. The collection of all \mathcal{C} -Cayley polynomial-time computable groups for all classes \mathcal{C} forms the family of *Cayley polynomial-time computable groups*. In Theorem 13 we show that the family of Cayley polynomial-time computable groups and the ones with quasigeodesic normal forms are each closed under taking f.g. subgroups. In the end of Section 4 we show that the class of REG-Cayley polynomial-time computable groups comprises all f.g. nilpotent groups, where REG is the class of regular languages. Moreover, it includes examples such as the wreath product $\mathbb{Z}_2 \wr \mathbb{Z}^2$ and Thompson's group F .

In Section 5 we study the Cayley distance function for Cayley p.f. linear-time computable and REG-Cayley polynomial-time computable representations. We demonstrate that some properties of the Cayley distance function which hold for Cayley automatic representations (shown in the previous works [4, 5]) do not hold neither for Cayley p.f. linear-time computable and nor for REG-Cayley polynomial-time computable representations. Section 6 concludes the paper. Figure 1 shows a Venn diagram for different classes of groups considered in this paper.

Related work. We briefly mention some previous works which extend the notion of an automatic group. The motivation to introduce such extensions was principally to include all fundamental groups of compact 3-manifolds. Bridson and Gilman introduced the notion of an asynchronously \mathcal{A} -combable group for an arbitrary class of languages \mathcal{A} [7]. Baumslag, Shapiro and Short introduced the class of parallel poly-pushdown groups [8]. Brittenham, Hemmiller and Holt introduced the notion of an autostackable group [9]. Kharlampovich, Khoushainov and Miasnikov introduced the notion of a Cayley automatic group [1] from which the present paper developed. The second author and Taback extended the notion of a Cayley automatic group replacing the class of regular languages with more powerful language classes [10], which we refine here.

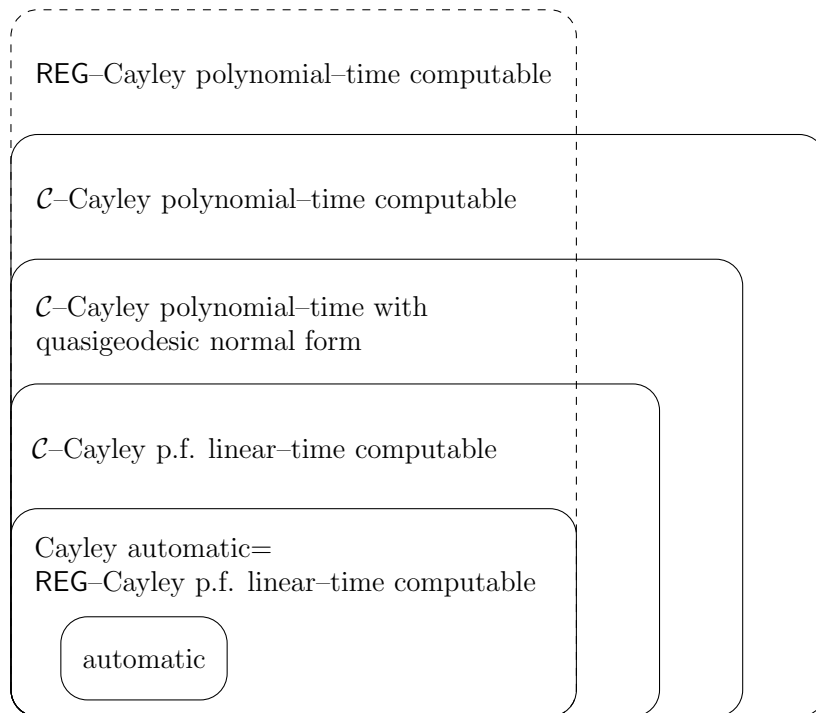


Figure 1: A Venn diagram of classes of interest.

2. Cayley Automatic Groups

Kharlampovich, Khousainov and Miasnikov introduced the notion of a Cayley automatic group³ [1] as a natural generalization of the notion of an automatic group [2] which uses the same computational model – a two-tape synchronous automaton. The class of Cayley automatic groups not only comprises all automatic groups, but it includes a rich a family of groups which are not automatic. In particular, it includes all f.g. nilpotent groups of nilpotency class two [1], the Baumslag–Solitar groups [11], higher rank lamplighter groups [12] and all wreath products of the form $G \wr H$, where G is Cayley automatic and H is virtually infinite cyclic [6]. We assume that the reader is familiar with the notion of a regular language, a finite automaton and a multi-tape synchronous automaton. Below we briefly recall

³A Cayley automatic group is also referred to as a Cayley graph automatic or graph automatic group in the literature.

both definitions: for automatic and Cayley automatic groups.

Let G be a finitely generated group with a generating set $A = \{a_1, \dots, a_n\} \subset G$. We denote by A^{-1} the set of the inverses of elements of A , that is, $A^{-1} = \{a_1^{-1}, \dots, a_n^{-1}\}$. Let $S = A \cup A^{-1}$. For a given word $w = a_{i_1}^{\sigma_1} \dots a_{i_m}^{\sigma_m} \in S^*$, where $i_1, \dots, i_m \in \{1, \dots, n\}$ and $\sigma_1, \dots, \sigma_m \in \{+1, -1\}$, let $\pi(w)$ be the product of elements $a_{i_1}^{\sigma_1} \dots a_{i_m}^{\sigma_m}$ in the group G ; if w is the empty string: $w = \epsilon$, then $\pi(w)$ is the identity of the group G . For a given language $L \subseteq S^*$, we denote by $\pi : L \rightarrow G$ the canonical map which sends a string⁴ $w \in L$ to the group element $\pi(w) \in G$.

It is said that the group G is automatic, if there is a regular language $L \subseteq S^*$ such that the canonical map $\pi : L \rightarrow G$ is bijective and for every $a \in A$ the relation $L_a = \{(u, v) \in L \times L \mid \pi(u)a = \pi(v)\}$ is recognized by a two-tape synchronous automaton. A string $w \in L$ is called a normal form for the group element $\pi(w) \in G$; accordingly, L is called a language of normal forms. We call the bijection $\pi : L \rightarrow G$ an automatic representation of the group G .

It is said that the group G is Cayley automatic if there is a regular language $L \subseteq \Sigma^*$ and a bijection $\psi : L \rightarrow G$ such that for every $a \in A$ the relation $R_a = \{(u, v) \in L \times L \mid \psi(u)a = \psi(v)\}$ is recognized by a two-tape synchronous automaton. Similarly, we say that L is a language of normal forms and a string $w \in L$ is a normal form for the group element $\psi(w) \in G$. We call the bijection $\psi : L \rightarrow G$ a Cayley automatic representation of the group G . We note that the notion of a Cayley automatic group does not require the bijection $\psi : L \rightarrow G$ to be canonical. As long as for every $a \in A$ the relation R_a is recognized by a two-tape synchronous automaton, ψ can be an arbitrary bijection. Similarly, as long as L is regular, it can be a language over an arbitrary alphabet Σ .

It is said that a function $f : \Sigma^* \rightarrow \Sigma^*$ is automatic if the relation $R_f = \{(w, f(w)) \in \Sigma^* \times \Sigma^* \mid w \in \Sigma^*\}$ is recognized by a two-tape synchronous automaton. So one can equivalently define Cayley automatic groups in the following way.

Definition 1 (Cayley automatic groups). *We say that the group G is Cayley automatic if there exists a regular language $L \subseteq \Sigma^*$ over some finite alphabet Σ , a bijective mapping $\psi : L \rightarrow G$ and automatic functions $f_s : \Sigma^* \rightarrow \Sigma^*$, $s \in S$, such that:*

⁴We use the terms “string” and “word” interchangeably.

- $f_s(L) \subseteq L$, that is, f_s maps a normal form to a normal form;
- for every $w \in L$: $\psi(f_s(w)) = \psi(w)s$, that is, the following diagram commutes:

$$\begin{array}{ccc} L & \xrightarrow{f_s} & L \\ \psi \downarrow & & \downarrow \psi \\ G & \xrightarrow{\times s} & G \end{array}$$

for all $s \in S$. We call $\psi : L \rightarrow G$ a Cayley automatic representation of G .

Remark 1. *The original motivation to study Cayley automatic groups stemmed not only from the notion of an automatic group [2], but also from the notion of a FA-presentable structure [13]. Namely, a f.g. group is Cayley automatic if and only if its labelled directed Cayley graph is a FA-presentable structure [1]. For a recent survey of the theory of FA-presentable structures we refer the reader to [14].*

3. Cayley position-faithful linear-time computable groups

The notion of a Cayley automatic group can be naturally extended further to a notion of a Cayley position-faithful linear-time computable group which we introduce in this section.

Let us first recall the notion of a position-faithful one-tape Turing machine (as defined in [3, p. 4]).

Definition 2 (Position-faithful one-tape Turing machine). *A position-faithful one-tape Turing machine is a Turing machine which uses a semi-infinite tape (infinite in one direction only) with the left-most position containing the special symbol \boxplus which only occurs at this position and cannot be modified. The initial configuration of the tape is $\boxplus x \square^\infty$, where \square is a special blank symbol, and $x \in \Sigma^*$ for some alphabet Σ with $\Sigma \cap \{\boxplus, \square\} = \emptyset$. During the computation the Turing machine operates as usual, reading and writing cells to the right of the \boxplus symbol.*

A function $f : \Sigma^ \rightarrow \Sigma^*$ is said to be computed by a position-faithful one-tape Turing machine, if when started with tape content being $\boxplus x \square^\infty$, the head initially being at \boxplus , the Turing machine eventually reaches an accepting state (and halts), with the tape content starting with $\boxplus f(x) \square$. There is no restriction on the output beyond the first appearance of \square .*

Case, Jain, Seah and Stephan established the equivalence of the following classes of functions [3]:

- automatic functions $f : \Sigma^* \rightarrow \Sigma^*$;
- functions $f : \Sigma^* \rightarrow \Sigma^*$ computed in linear time by a deterministic position-faithful one-tape Turing machine.
- functions $f : \Sigma^* \rightarrow \Sigma^*$ computed in linear time by a nondeterministic position-faithful one-tape Turing machine.

We say that a function $f : \Sigma^* \rightarrow \Sigma^*$ is position-faithful (p.f.) linear-time computable if it is computed by a (deterministic) position-faithful one-tape Turing machine in linear time. By the equivalence above $f : \Sigma^* \rightarrow \Sigma^*$ is p.f. linear-time computable if and only if it is automatic. So we may use the terms automatic and p.f. linear-time computable interchangeably.

We note that the requirements of being one-tape and position-faithful matter. Consider the following example from [3, p. 4]: a function which takes input $w \in \Sigma$ and outputs the binary string $v \in \{0, 1\}^*$ where $w = uvxy$ with $u \in (\Sigma \setminus \{0, 1\})^*$, $x \in \Sigma \setminus \{0, 1\}$. An ordinary semi-infinite tape Turing machine can easily compute this: simply move the read head to the first occurrence of 0, 1 on the tape (or replace all cells u, x, y by blank symbols, depending on the Turing machine model). The position-faithful model is not able to perform this function in linear time: it would have to somehow copy the contents of the cells containing v forwards so that they start after the \boxplus symbol, but this would involve at least $O(|u|^2)$ steps. Note also that this function can be computed by a deterministic position-faithful *two-tape* Turing machine in linear time. The functions computed by position-faithful one-tape Turing machines are in a certain sense a small natural class of linear time computable functions.

In order to make our first generalisation of Cayley automatic, we do not require any longer a language of normal forms to be regular like we do for Cayley automatic groups. But we require the right multiplication by a generator, or its inverse, to be computed by an automatic function. Let G be a f.g. group and $S = \{s_1, \dots, s_n\} \subset G$ be a finite set of semigroup generators of G : that is, every $g \in G$ can be represented as a product of elements from S . Let \mathcal{C} be a nonempty class of languages.

Definition 3 (*\mathcal{C} -Cayley position-faithful linear-time computable group*).
We say that the group G is \mathcal{C} -Cayley position-faithful (p.f.) linear-time

computable group if there exist a language $L \subseteq \Sigma^*$ from the class \mathcal{C} over some finite alphabet Σ , a bijective mapping $\psi : L \rightarrow G$ between the language L and the group G and p.f. linear-time computable functions $f_i : \Sigma^* \rightarrow \Sigma^*$ such that $f_i(L) \subseteq L$ and for every $w \in L$: $\psi(f_i(w)) = \psi(w)s_i$, for all $i = 1, \dots, n$. We call $\psi : L \rightarrow G$ a \mathcal{C} -Cayley p.f. linear-time computable representation of the group G . If the requirement for L to be in a specific class \mathcal{C} is omitted, then we say that G is a Cayley p.f. linear-time computable group and $\psi : L \rightarrow G$ is a Cayley p.f. linear-time computable representation of G .

The class of REG-Cayley p.f. linear-time computable groups is simply the class of Cayley automatic groups. Below we show that, similarly to Cayley automatic groups, Definition 3 does not depend on the choice of generators.

Proposition 1. *The notion of a \mathcal{C} -Cayley p.f. linear-time computable group does not depend on the choice of generators.*

Proof. Let G be a \mathcal{C} -Cayley p.f. linear-time computable group for a set of semigroup generators $S = \{s_1, \dots, s_n\}$. Then there is a language $L \subseteq \Sigma^*$ from the class \mathcal{C} , a bijective mapping $\psi : L \rightarrow G$ and automatic functions $f_i : \Sigma^* \rightarrow \Sigma^*$ such that $f_i(L) \subseteq L$ and $\psi(f_i(w)) = \psi(w)s_i$ for all $i = 1, \dots, n$ and $w \in L$. Let $S' = \{s'_1, \dots, s'_k\} \subseteq G$ be another set of semigroup generators of the group G . Each element $s' \in S'$ is a product of elements from S . Therefore, for a given $j = 1, \dots, k$ there exist $s_{j_1}, \dots, s_{j_m} \in S$ for which $s'_j = s_{j_1} \dots s_{j_m}$. We define $f'_j : \Sigma^* \rightarrow \Sigma^*$ to be the composition: $f'_j = f_{j_m} \circ \dots \circ f_{j_1}$. For every $j = 1, \dots, k$ the function f'_j is automatic, $f'_j(L) \subseteq L$ and $\psi(f'_j(w)) = \psi(w)s'_j$ for all $w \in L$. This shows that the definition of the class of \mathcal{C} -Cayley p.f. linear-time computable groups does not depend on the choice of semigroup generators S . \square

Remark 2. *By Proposition 1 one can always assume that a set of semigroup generators S is symmetric. That is, $S = A \cup A^{-1}$ for some finite set A generating G , where A^{-1} is the set of inverses of elements from A : $A^{-1} = \{a^{-1} \mid a \in A\}$.*

Remark 3. *Similarly to Cayley automatic groups, Cayley p.f. linear-time computable groups are related to the notion of a FA-presentable structure. Let \mathfrak{B} be the structure $\mathfrak{B} = (\Sigma^*; \text{Graph}(f_1), \dots, \text{Graph}(f_n))$ for some Σ and f_1, \dots, f_n from Definition 3, where $\text{Graph}(f)$ for a function $f : \Sigma^* \rightarrow \Sigma^*$ is the binary relation $\text{Graph}(f) = \{(w, f(w)) \in \Sigma^* \times \Sigma^* \mid w \in \Sigma^*\}$. Then*

every structure $\mathfrak{B}' = (B'; f'_1, \dots, f'_n)$ isomorphic to the structure \mathfrak{B} is FA-presentable. Let Γ be the directed labelled graph $\Gamma = (G; E_1, \dots, E_n)$, where $E_i = \{(g_1, g_2) \in G \times G \mid g_1 s_i = g_2\}$. Then the bijection $\psi^{-1} : G \rightarrow L$ is an embedding of the structure Γ into the structure \mathfrak{B} .

3.1. Quasigeodesic Normal Form

We notice that the analogue of the bounded difference lemma (see [2, Lemma 2.3.9] for automatic and [1, Lemma 8.1] for Cayley automatic groups) holds for Cayley p.f. linear-time computable groups as well. Let G be a Cayley p.f. linear-time computable group with a set of generators $A \subset G$ and $\psi : L \rightarrow G$ be a Cayley p.f. linear-time computable representation of G for some language $L \subseteq \Sigma^*$.

Lemma 1. *There exists a constant $K > 0$ such that for every $g \in G$ and $s \in A \cup A^{-1}$, if $u, v \in L$ are the strings representing g and gs , respectively: $\psi(u) = g$ and $\psi(v) = gs$, then $\|u\| - \|v\| \leq K$.*

Proof. For every $s \in A \cup A^{-1}$ there is an automatic function $f_s : \Sigma^* \rightarrow \Sigma^*$ such that $f_s(u) = v$ for all $u, v \in L$ for which $\psi(u)s = \psi(v)$ in the group G . For a given $s \in A \cup A^{-1}$, let M_s be a (nondeterministic) two-tape synchronous automaton recognizing the relation $R_{f_s} = \{(w, f_s(w)) \in \Sigma^* \times \Sigma^* \mid w \in \Sigma^*\}$. By the pumping lemma, for every $(u, v) \in R_{f_s}$ the following inequality holds:

$$\|v\| \leq \|u\| + N_s,$$

where N_s is the number of states of the automaton M_s . Therefore, for all $u, v \in L$, if $\psi(u) = g$ and $\psi(v) = gs$ for some $g \in G$ and $s \in A \cup A^{-1}$, then $\|u\| - \|v\| \leq K$, where $K = \max\{N_s \mid s \in A \cup A^{-1}\}$. \square

For a given group element $g \in G$ we denote by $d_A(g)$ the length of a geodesic word representing g with respect to the set of generators A .

Theorem 1 (Quasigeodesic normal form). *For a given Cayley p.f. linear-time computable representation $\psi : L \rightarrow G$ and a set of generators A , there exists a constant C such that for every $w \in L$ the following inequality holds:*

$$\|w\| \leq C(d_A(\psi(w)) + 1). \quad (1)$$

Proof. For a given $w \in L$, let $a_1 \dots a_n$, for $a_i \in A \cup A^{-1}, i = 1, \dots, n$, be a geodesic in G with respect to the set of generators A such that $a_1 \dots a_n =$

$\psi(w)$ in G ; so, $d_A(\psi(w)) = n$. We denote by w_0 the string representing the identity e : $\psi(w_0) = e$. For a given $i \in \{1, \dots, n\}$, let $w_i = a_1 \dots a_i$. By Lemma 1, $|w_{i+1}| \leq |w_i| + K$ for all $i = 0, \dots, n-1$ and some constant K . Therefore, $|w| \leq nK + |w_0|$. Let $C = \max\{K, |w_0|\}$. Thus, $|w| \leq C(d_A(\psi(w)) + 1)$. \square

3.2. Algorithmic Properties

A key property of Cayley automatic groups which they share with automatic groups is the existence of a quadratic time algorithm which for a given word $v \in (A \cup A^{-1})^*$ finds the normal form $u \in L$, i.e., the string for which $\psi(u) = \pi(v)$; see [2, Theorem 2.3.10] and [1, Theorem 8.2] for automatic and Cayley automatic groups, respectively. Below we show that this property holds for Cayley p.f. linear-time computable groups as well.

Theorem 2 (Computing normal form in quadratic time). *There is an algorithm which for a given input word $v \in (A \cup A^{-1})^*$ computes the string $u \in L$, for which $\psi(u) = \pi(v)$ in the group G . Moreover, this algorithm can be implemented by a deterministic position-faithful one-tape Turing machine in quadratic time.*

Proof. Let us be given the string $u_0 \in L$ representing the identity $e \in G$: $\psi(u_0) = e$. Let $v = b_1 \dots b_k$, where $b_i \in A \cup A^{-1}$. For a given $i = 1, \dots, k$ we denote by TM_{b_i} a position-faithful deterministic one-tape Turing machine which computes the function f_{b_i} in linear time. An algorithm which finds the representative string $u \in L$ for the input word v works as follows. First it computes the representative $u_1 \in L$ of b_1 by feeding u_0 to TM_{b_1} as the input. Then it feeds the string u_1 to TM_{b_2} as the input to obtain the representative $u_2 \in L$ of the group element $b_1 b_2$. Continuing in this way it computes the representative $u_k \in L$ of the group element $b_1 \dots b_k$. By Lemma 1, for every $u_j \in L, j = 1, \dots, k$: $|u_j| \leq |u_0| + Kj$. Moreover, there are constants C_1, C_0 such that for every $j = 1, \dots, k$ the Turing machine TM_{b_j} computes u_j from the input u_{j-1} in time at most $C_1|u_{j-1}| + C_0 \leq C_1(|u_0| + K(j-1)) + C_0$. If the head of a Turing machine points at the initial cell then $2(j-1)$ moves are required to read off the symbol b_j and return the head back to the initial cell. Thus, the time required to compute the representative $u = u_k$ is bounded by $C_2 k^2$ for some constant C_2 . Clearly, this algorithm can be implemented by a position-faithful one-tape deterministic Turing machine for which initially the word v is written on a tape with the head pointing at the initial cell containing the symbol b_1 . \square

Corollary 1 (Solving word problem in quadratic time). *For a Cayley p.f. linear-time computable group the word problem can be solved by a deterministic one-tape Turing machine in quadratic time.*

Proof. An algorithm solving the word problem in G is as follows. For a given input word $v \in (A \cup A^{-1})^*$ it first finds the string $u \in L$ representing $\pi(v)$: $\psi(u) = \pi(v)$, as it is described in Theorem 2, and then compares u with the string u_0 representing the identity $e \in G$: if $u = u_0$, then $\pi(v) = e$; otherwise, $\pi(v) \neq e$. This algorithm can be implemented by a deterministic one-tape Turing machine. \square

Theorem 3. *Let RE denote the class of recursively enumerable languages. For every Cayley p.f. linear-time computable representation $\psi : L \rightarrow G$ the language L is in the class RE.*

Proof. A procedure listing all words of the language L is as follows. It consecutively takes $v \in (A \cup A^{-1})^*$ as the input to produce the output $\psi^{-1}(v) \in L$ using the algorithm described in Theorem 2. This procedure lists all strings of the language L . \square

Proposition 2. *Let R denote the class recursive languages. The class of $(\text{RE} \setminus \text{R})$ -Cayley p.f. linear-time computable groups is non-empty.*

Proof. There exists a f.g. subgroup $H \leq F_2 \times F_2$ with undecidable membership problem [15]: given a word w over some generating set of $F_2 \times F_2$, decide whether $\pi(w)$ is an element of H . Let $\psi : L \rightarrow F_2 \times F_2$ be a Cayley p.f. linear-time computable representation of $F_2 \times F_2$ (e.g., it can be Cayley automatic or even automatic one). Let $L' = \psi^{-1}(H)$ and $\psi' : L' \rightarrow H$ be the restriction of ψ onto L' : $\psi' = \psi|_{L'}$. By Theorem 7 below, $\psi' : L' \rightarrow H$ is a Cayley p.f. linear-time computable representation of H . If L' is recursive, then the algorithm solving membership problem for H is as follows. For a given word w over some generating set of $F_2 \times F_2$ we first find the string $u \in L$ for which $\psi(u) = \pi(w)$ in $F_2 \times F_2$ (see the algorithm in Theorem 2) and then verify whether u is in the language L' or not. Therefore, assuming that L' is recursive, we get that the membership problem for the subgroup $H \leq F_2 \times F_2$ must be decidable, which leads to a contradiction. Therefore, L' is not a recursive language, although it is recursively enumerable by Theorem 3. \square

3.3. Closure Properties

Now we turn to closure properties for Cayley p.f. linear-time computable groups. Let \mathcal{C} be a given class of languages. Throughout the paper we assume that \mathcal{C} is closed under a change of symbols in the alphabet. That is, if $\xi : \Sigma \rightarrow \Sigma'$ is a bijection between two finite alphabets Σ and Σ' and L is in the class \mathcal{C} , then the image of L under the homomorphism induced by ξ is also in the class \mathcal{C} .

Theorem 4 (Finite extensions). *Assume that a class of languages \mathcal{C} satisfies the following closure property: if $L \subseteq \Sigma^*$ is in the class \mathcal{C} and L_0 is a finite language over some Σ_0 for which $\Sigma \cap \Sigma_0 = \emptyset$, then the concatenation LL_0 is in the class \mathcal{C} . Then, a finite extension of a \mathcal{C} -Cayley p.f. linear-time computable group is \mathcal{C} -Cayley p.f. linear-time computable.*

Proof. Let H be a subgroup of finite index of a group G . Suppose that H is \mathcal{C} -Cayley p.f. linear-time computable. Then there exists a \mathcal{C} -Cayley p.f. linear-time computable representation $\psi : L \rightarrow H$ for some language $L \subseteq \Sigma^*$ in the class \mathcal{C} . The following is similar to the argument from [1, Theorem 10.1] which shows that a finite extension of Cayley automatic group is Cayley automatic. Every $g \in G$ is uniquely represented as a product $g = hk$, where $h \in H$ and $k \in K$ for some finite subset $K = \{k_0, k_1, \dots, k_m\} \subset G$ that contains the identity: $k_0 = e \in G$. Let $\Sigma_0 = \{\sigma_1, \dots, \sigma_m\}$ for some symbols $\sigma_1, \dots, \sigma_m$, which are not in Σ , and $L_0 \subset \Sigma_0^*$ be a finite language $L_0 = \{\epsilon, \sigma_1, \dots, \sigma_m\}$. We denote by $L' = LL_0$ the concatenation of the languages L and L_0 . By the assumption of the theorem, the language L' is in the class \mathcal{C} . Let $A = \{a_1, \dots, a_n\}$ be some set of generators of H . Then $A \cup A^{-1} \cup K$ is a set of semigroup generators for G .

We define a bijection $\psi' : L' \rightarrow G$ as follows. For a given $w' \in L'$, w' is the concatenation: $w' = wu$ for some $w \in L$ and $u \in L_0$. Let $\varphi : L_0 \rightarrow K$ be a bijection between L_0 and K for which $\varphi(\epsilon) = k_0, \varphi(\sigma_1) = k_1, \dots, \varphi(\sigma_m) = k_m$. We put $\psi'(w') = \psi(w)\varphi(u)$. The right multiplication of $g \in G$ by $q \in A \cup A^{-1} \cup K$ is given by the formula:

$$gq = h k q = h s_1 \dots s_\ell k',$$

for some $s_1, \dots, s_\ell \in A \cup A^{-1}$ and $k_j \in K$ which depend only on k and q : $kq = s_1 \dots s_\ell k_j$. An algorithm transforming the input $\psi'^{-1}(g)$ to the output $\psi'^{-1}(gq)$, implemented by a position-faithful one-tape Turing machine, is as follows. First the head moves to the rightmost cell which contains the

symbol $\varphi^{-1}(k)$ (or blank symbol if $k = e$), reads it off, stores it in the memory and changes it to the blank symbol; then the head moves back to the initial cell. Now the string $\psi^{-1}(h)$ is written on the tape. After that an algorithm computing multiplication by $s_1 \dots s_\ell$ in the group H is run; once it is finished, the string $\psi^{-1}(hs_1 \dots s_\ell)$ is written on the tape. Then the head moves to the first blank symbol to change it to $\varphi^{-1}(k')$, unless $k' = e$ – in this case no action is needed. Then this Turing machine halts. Now the string $\psi^{-1}(hs_1 \dots s_\ell)\varphi^{-1}(k')$, which is equal to $\psi'^{-1}(gq)$, is written on the tape. Clearly, at most linear time is required for this algorithm. Thus, G is \mathcal{C} -Cayley p.f. linear-time computable. \square

Theorem 5 (Direct products). *Assume that a class of languages \mathcal{C} satisfies the following closure property: if $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$ are some languages in the class \mathcal{C} for which $\Sigma_1 \cap \Sigma_2 = \emptyset$, then the concatenation L_1L_2 is in the class \mathcal{C} . Then, the direct product of two \mathcal{C} -Cayley p.f. linear-time computable groups is \mathcal{C} -Cayley p.f. linear-time computable.*

Proof. Let G_1 and G_2 be two \mathcal{C} -Cayley p.f. linear-time computable groups. Then there exist \mathcal{C} -Cayley p.f. linear-time computable representations $\psi_1 : L_1 \rightarrow G_1$ and $\psi_2 : L_2 \rightarrow G_2$ for some languages $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$ in the class \mathcal{C} for which $\Sigma_1 \cap \Sigma_2 = \emptyset$. We denote by L the concatenation: $L = L_1L_2$. By the assumption of the theorem the language L is in the class \mathcal{C} . Let $A = \{a_1, \dots, a_{n_1}\}$ and $B = \{b_1, \dots, b_{n_2}\}$ be some sets of generators for the groups G_1 and G_2 , respectively. Then $A \cup A^{-1} \cup B \cup B^{-1}$ is a set of semigroup generators for the group $G = G_1 \times G_2$. The groups G_1 and G_2 can be considered as subgroups of G . Every group element $g \in G$ can be uniquely represented as the product: $g = g_1g_2$, where $g_1 \in G_1$ and $g_2 \in G_2$.

Let $\psi : L \rightarrow G$ be a bijection defined as follows. For a given $w \in L$, let w be the concatenation $w = uv$ for some $u \in L_1$ and $v \in L_2$. We put $\psi(w) = \psi_1(u)\psi_2(v)$. The right multiplication of $g = g_1g_2$, where $g_1 \in G_1$ and $g_2 \in G_2$, by $q \in A \cup A^{-1} \cup B \cup B^{-1}$ is given by $gq = (g_1q)g_2$ if $q \in A \cup A^{-1}$ and $gq = g_1(g_2q)$ if $q \in B \cup B^{-1}$. For the case $q \in A \cup A^{-1}$, an algorithm transforming the input $\psi^{-1}(g) = \psi_1^{-1}(g_1)\psi_2^{-1}(g_2)$ to the output $\psi^{-1}(gq)$, implemented by a position-faithful one-tape Turing machine, is as follows. First it transforms the prefix $\psi_1^{-1}(g_1)$ to the prefix $\psi_1^{-1}(g_1q)$. If overlapping with the substring $\psi_2^{-1}(g_2)$ occurs, then it can be encoded by special symbols: for example, if symbols A and B appear on the same cell of the tape, then it can be encoded by the symbol $A \otimes B$. After that the algorithm shifts the substring $\psi_2^{-1}(g_2)$ either to the left or to the right, so it

is written right after the string $\psi_1^{-1}(g_1q)$. By Lemma 1, only shifting (left or right) by a finite number of cells is needed. Therefore, at most linear time is required for our algorithm. If $q \in B \cup B^{-1}$, an algorithm just updates the suffix $\psi_2^{-1}(g_2)$ to the suffix $\psi_2^{-1}(g_2q)$ while the prefix $\psi_1^{-1}(g_1)$ remains unchanged. Clearly, at most linear time is needed for this algorithm to be implemented by a one-tape position-faithful Turing machine. Finally we conclude that G is a \mathcal{C} -Cayley p.f. linear-time computable group. \square

Theorem 6 (Free products). *Assume that a class of languages \mathcal{C} satisfies the following closure properties:*

- (a) *if a nonempty language L is in the class \mathcal{C} and $\epsilon \notin L$, then for every $w \in L$ the language $(L \setminus \{w\}) \vee \{\epsilon\}$ is in the class \mathcal{C} ;*
- (b) *if $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$ are some languages $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$ in the class \mathcal{C} which contain the empty string $\epsilon \in L_1, L_2$ and for which $\Sigma_1 \cap \Sigma_2 = \emptyset$, then the language $L = (L'_1 L'_2)^* \vee (L'_1 L'_2)^* L'_1 \vee (L'_2 L'_1)^* \vee (L'_2 L'_1)^* L'_2 \vee \{\epsilon\}$ is in the class \mathcal{C} , where $L'_1 = L_1 \setminus \{\epsilon\}$ and $L'_2 = L_2 \setminus \{\epsilon\}$.*

Then, the free products of two \mathcal{C} -Cayley p.f. linear-time computable groups is \mathcal{C} -Cayley p.f. linear-time computable.

Proof. Let G_1, G_2 be \mathcal{C} -Cayley p.f. linear-time computable groups. There exist \mathcal{C} -Cayley p.f. linear-time computable representations $\psi_1 : L_1 \rightarrow G_1$ and $\psi_2 : L_2 \rightarrow G_2$ for some languages $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$ in the class \mathcal{C} for which $\Sigma_1 \cap \Sigma_2 = \emptyset$. Suppose that $\epsilon \in L_1$ and for some string $w \in L_1$, $w \neq \epsilon$: $\psi_1(w) = e$ in G_1 . Let $\psi'_1 : L_1 \rightarrow G_1$ be a bijective map for which $\psi'_1(u) = \psi_1(u)$ for all $u \in L_1 \setminus \{\epsilon, w\}$ and $\psi'_1(w) = \psi_1(\epsilon)$ and $\psi'_1(\epsilon) = e$. It can be easily seen that $\psi'_1 : L_1 \rightarrow G_1$ is a \mathcal{C} -Cayley p.f. linear-time computable representation. Now suppose that $\epsilon \notin L_1$. Let w be a word from L_1 for which $\psi_1(w) = e$. By the property (a), the language $L''_1 = (L_1 \setminus \{w\}) \vee \{\epsilon\}$ is in the class \mathcal{C} . Let $\psi''_1 : L''_1 \rightarrow G_1$ be a bijective map for which $\psi''_1(u) = \psi_1(u)$ for all $u \in L_1 \setminus \{w\}$ and $\psi''_1(\epsilon) = e$. Then ψ''_1 is a \mathcal{C} -Cayley p.f. linear-time computable representation. Thus we can always assume that $\epsilon \in L_1$ and $\psi_1(\epsilon) = e$ in G_1 . We assume the same for L_2 and ψ_2 : $\epsilon \in L_2$ and $\psi_2(\epsilon) = e$ in G_2 .

The groups G_1 and G_2 are naturally embedded in the free product $G = G_1 \star G_2$, so we consider them as the subgroups of G . Now let $L = (L'_1 L'_2)^* \vee (L'_1 L'_2)^* L'_1 \vee (L'_2 L'_1)^* \vee (L'_2 L'_1)^* L'_2 \vee \{\epsilon\}$, where $L'_1 = L_1 \setminus \{\epsilon\}$ and $L'_2 =$

$L_2 \setminus \{\epsilon\}$. By the property (b), the language L is in the class \mathcal{C} . We define a bijection $\psi : L \rightarrow G$ as follows. We put $\psi(\epsilon) = e$. For $w = u_1v_1 \dots u_nv_n \in (L'_1L'_2)^*$, where $u_i \in L'_1$ and $v_i \in L'_2$ for $i = 1, \dots, n$, we put: $\psi(w) = \psi_1(u_1)\psi_2(v_1) \dots \psi_1(u_n)\psi_2(v_n)$. For $w \in (L'_1L'_2)^*L'_1, (L'_2L'_1)^*$ and $(L'_2L'_1)^*L'_2$, $\psi(w)$ is defined in a similar way. Let $A = \{a_1, \dots, a_{n_1}\}$ and $B = \{b_1, \dots, b_{n_2}\}$ be some sets of generators for the groups G_1 and G_2 , respectively. Then $A \cup A^{-1} \cup B \cup B^{-1}$ is a set of semigroup generators for the group $G = G_1 \star G_2$. For a given $g \in G$ let us assume that $\psi^{-1}(g) = u_1v_1 \dots u_nv_n \in (L'_1L'_2)^*$. If $q \in B \cup B^{-1}$, an algorithm transforming the input $\psi^{-1}(g)$ to the output $\psi^{-1}(gq)$ updates the suffix v_n to the suffix $\psi_2^{-1}(\psi_2(v_n)q)$ while the prefix $u_1v_1 \dots v_{n-1}u_n$ is left unchanged. If $q \in A \cup A^{-1}$, an algorithm simply attaches the string $\psi_1^{-1}(q)$ to $\psi^{-1}(g)$ as a suffix. For $\psi^{-1}(g) \in (L'_1L'_2)^*L'_1 \vee (L'_2L'_1)^* \vee (L'_2L'_1)^*L'_2$ an algorithm transforming $\psi^{-1}(g)$ to $\psi^{-1}(gq)$ is implemented in a similar way. The case $g = e$ is trivial. Clearly, this algorithm is implemented by a one-tape position-faithful Turing machine in at most linear time. Thus, the group G is \mathcal{C} -Cayley p.f. linear-time computable. \square

Remark 4. *We note that the conditions imposed on the class \mathcal{C} in Theorems 4, 5 and 6 are weak. Obviously, these conditions are satisfied for many classes of languages including, e.g., regular, (deterministic) context-free, (deterministic) context-sensitive, recursive, k -counter, k -context-free.*

Theorem 7 (Finitely generated subgroups). *A finitely generated subgroup of a Cayley p.f. linear-time computable group is Cayley p.f. linear-time computable.*

Proof. Let G be a Cayley p.f. linear-time computable group and $S = A \cup A^{-1} = \{s_1, \dots, s_n\}$ be a set of semigroup generators of G . Then there is a Cayley p.f. linear-time computable representation $\psi : L \rightarrow G$ for some language $L \subseteq \Sigma^*$. Let $H \leq G$ be a finitely generated subgroup of G and $S' = A' \cup A'^{-1} = \{s'_1, \dots, s'_k\}$ be a set of semigroup generators of H . Let $L' = \psi^{-1}(H) \subset L$. We define $\psi' : L' \rightarrow H$ as the restriction of ψ onto L' : for a given $w \in L'$, $\psi'(w) = \psi(w)$. In order to prove that the representation $\psi' : L' \rightarrow H$ is Cayley p.f. linear-time computable we repeat the argument from Proposition 1. Let $f_i : \Sigma^* \rightarrow \Sigma^*$ be automatic functions corresponding to multiplications in G by the semigroup generators s_i , for $i = 1, \dots, n$ respectively: $\psi(f_i(w)) = \psi(w)s_i$ for all $w \in L$. For a given $j = 1, \dots, k$ there exist j_1, \dots, j_m for which $s'_j = s_{j_1} \dots s_{j_m}$. For every $j = 1, \dots, k$ the function $f'_j = f_{j_m} \circ \dots \circ f_{j_1}$ is automatic, $f'_j(L') \subseteq L'$ and $\psi'(f'_j(w)) = \psi'(w)s'_j$ for all $w \in L'$. Therefore, the group H is Cayley p.f. linear-time computable. \square

Remark 5. We remark that the language L' in the proof of Theorem 7 is not necessarily in the same class as the language L . An illustrative example, when L is a regular language but L' is not recursive, is shown in Proposition 2.

3.4. Relation with \mathcal{C} -graph Automatic Groups

In order to extend the class of Cayley automatic groups, the second author and Taback introduced the notion of a $(\mathcal{B}, \mathcal{C})$ -graph automatic group [10]. Let G be a group, S be a symmetric generating set of G and Σ be a finite alphabet. A tuple (G, S, Σ) is said to be $(\mathcal{B}, \mathcal{C})$ -graph automatic if there is a bijection $\psi : L \rightarrow G$ between a language $L \subseteq \Sigma^*$ from the class \mathcal{B} and a group G such that for every $s \in S$ the language $L_s = \{u \otimes v \mid u, v \in L, \psi(u)s = \psi(v)\}$ is in the class \mathcal{C} . If $\mathcal{B} = \mathcal{C}$, then the tuple (G, S, Σ) is said to be \mathcal{C} -graph automatic.

Theorem 8. Assume that a class of languages \mathcal{C} satisfies the following properties:

- (a) if $L \subseteq \Sigma^*$ is some language in the class \mathcal{C} , then $L \otimes \Sigma^* = \{u \otimes v \mid u \in L, v \in \Sigma^*\}$ is in the class \mathcal{C} ;
- (b) if R is a regular language and L is in the class \mathcal{C} , then $R \cap L$ is in the class \mathcal{C} .

Then, for a given \mathcal{C} -Cayley p.f. linear-time computable group G , the tuple (G, S, Σ) is \mathcal{C} -graph automatic for some alphabet Σ and every symmetric generating set S .

Proof. Let G be a \mathcal{C} -Cayley p.f. linear-time computable group for some class \mathcal{C} satisfying the conditions (a) and (b) of the theorem. Then there exists a \mathcal{C} -Cayley p.f. linear-time computable representation $\psi : L \rightarrow G$ for some language $L \subseteq \Sigma^*$ in the class \mathcal{C} . By the condition (a), the language $L \otimes \Sigma^*$ is in the class \mathcal{C} . Let A be a set of generators of G . For a given semigroup generator $s \in S = A \cup A^{-1}$ there exists an automatic function $f_s : \Sigma^* \rightarrow \Sigma^*$ such that $f_s(L) \subseteq L$ and $\psi(f_s(w)) = \psi(w)s$ for all $w \in L$. Since f_s is automatic, the language $R_s = \{u \otimes f_s(u) \mid u \in \Sigma^*\} \subseteq \Sigma^* \otimes \Sigma^*$ is regular. Therefore, by the condition (b), the language $(L \otimes \Sigma^*) \cap R_s$ is in the class \mathcal{C} . Thus, for every $s \in S$ the language $\{u \otimes v \mid u, v \in L, \psi(u)s = \psi(v)\} = (L \otimes \Sigma^*) \cap R_s$ is in the class \mathcal{C} , so (G, S, Σ) is \mathcal{C} -graph automatic. \square

Remark 6. We note that the condition imposed on the class \mathcal{C} in Theorem 8 is satisfied for a wide family of languages including all those mentioned in Remark 4.

Remark 7. We denote by DCS the class of deterministic context-sensitive languages. By [10, Theorem 15], a finitely generated group G is DCS-graph automatic with quasigeodesic normal form if and only if it is a group with DCS word problem. Therefore, by Theorems 1 and 8, a DCS-Cayley p.f. linear-time computable group is a group with DCS word problem.

3.5. Examples

Thurston proved that an automatic nilpotent group must be virtually abelian [2]. Kharlampovich, Khoushainov and Miasnikov showed that every f.g. nilpotent group of nilpotency class at most two is Cayley automatic [1]. However, it is conjectured that there exists a f.g. nilpotent group of nilpotency class three which is not Cayley automatic [16]. The main purpose of this subsection is to show that Cayley p.f. linear-time computable groups comprise a wide family of groups including all f.g. subgroups of $\mathrm{GL}(n, \mathbb{Q})$. This implies that all polycyclic groups are Cayley p.f. linear-time computable. The latter, in particular, shows that all f.g. nilpotent groups are Cayley p.f. linear-time computable. The groups $\mathrm{SL}(n, \mathbb{Z})$ are also Cayley p.f. linear-time computable.⁵

Theorem 9. A finitely generated subgroup of $\mathrm{GL}(n, \mathbb{Q})$ is Cayley p.f. linear-time computable.

Proof. Let G be a f.g. subgroup of $\mathrm{GL}(n, \mathbb{Q})$ and S be a set of semigroup generators of G . Each $s \in S$ corresponds to a matrix $M_s \in \mathrm{GL}(n, \mathbb{Q})$ with rational coefficients $m_{s,ij} = \frac{p_{s,ij}}{q_{s,ij}}$ for $i, j = 1, \dots, n$, where $p_{s,ij}, q_{s,ij} \in \mathbb{Z}$ and $q_{s,ij} > 0$. Now we notice that there exist an integer $k > 0$ and integers $r_{s,ij}$ such that $m_{s,ij} = \frac{r_{s,ij}}{k}$ for all $s \in S$ and $i, j = 1, \dots, n$; for example, one can put $k = \prod_{s \in S} \prod_{i,j=1}^n q_{s,ij}$. Therefore, we may assume that for all $s \in S$ and $i, j = 1, \dots, n$: $m_{s,ij} \in \mathbb{Z}[1/k]$, where $\mathbb{Z}[1/k]$ is the abelian group of all rational numbers of the form $\frac{d}{k^\ell}$ for $d, \ell \in \mathbb{Z}$ and $\ell \geq 0$. For example, if $k = 10$,

⁵We recall that $\mathrm{SL}(2, \mathbb{Z})$ is automatic; so, it is also Cayley automatic. It is not known whether the groups $\mathrm{SL}(n, \mathbb{Z})$ for $n > 2$ are Cayley automatic or not.

then $\mathbb{Z}[1/k]$ is just the group of all finite fractional decimal numbers, i.e., the rational numbers for which the number of digits after the dot is finite. Since all coefficients of the matrices M_s , $s \in S$ are in $\mathbb{Z}[1/k]$, then for every matrix from G the coefficients of this matrix are also in $\mathbb{Z}[1/k]$. That is, G consists of matrices with coefficients from $\mathbb{Z}[1/k]$. Therefore, $G \subset M_n(\mathbb{Z}[1/k])$, where $M_n(\mathbb{Z}[1/k])$ is the ring of $n \times n$ matrices with coefficients in $\mathbb{Z}[1/k]$.

The abelian group $(\mathbb{Z}[1/k], +)$ is FA-presentable, see the proof, e.g., in [17]. If $k = 10$, then one can simply use the standard decimal representation of numbers from $\mathbb{Z}[1/k]$. For other values of k , one can use a representation in base k . Let us choose any FA-presentation of $(\mathbb{Z}[1/k], +)$, i.e., a bijection $\varphi : L_1 \rightarrow \mathbb{Z}[1/k]$ from some regular language L_1 to $\mathbb{Z}[1/k]$ for which the relation $R_+ = \{(u, v, w) \in L_1 \times L_1 \times L_1 \mid \varphi(u) + \varphi(v) = \varphi(w)\}$ is FA-recognizable. The latter also implies that multiplication by any fixed number $t = \frac{p}{k^i} \in \mathbb{Z}[1/k]$ is FA-recognizable. That is, the relation $R_t = \{(u, v) \in L_1 \times L_1 \mid \varphi(u)t = \varphi(v)\}$ is FA-recognizable. Now, every matrix $C \in M_n(\mathbb{Z}[1/k])$ with coefficients $c_{ij} \in \mathbb{Z}[1/k]$ for $i, j = 1, \dots, n$ we represent as the convolution $\varphi^{-1}(c_{11}) \otimes \varphi^{-1}(c_{12}) \otimes \dots \otimes \varphi^{-1}(c_{nn})$. The collection of all such convolutions form a regular language $L_n = \{u_{11} \otimes \dots \otimes u_{nn} \mid u_{ij} \in L_1, i, j = 1, \dots, n\}$. This gives the bijection $\varphi_n : L_n \rightarrow M_n(\mathbb{Z}[1/k])$ between L_n and $M_n(\mathbb{Z}[1/k])$.

For a given matrix C , the result of the multiplication of C by a matrix M_s for $s \in S$ is given by the following: for given $i, j = 1, \dots, n$, the coefficient d_{ij} of the matrix $D = CM_s$ equals $d_{ij} = c_{i1}m_{s,1j} + \dots + c_{in}m_{s,nj}$. Therefore, since R_+ and R_t for all $t \in \mathbb{Z}[1/k]$ are FA-recognizable, the relation $R_s = \{(u, v) \in L_n \times L_n \mid \varphi_n(u)M_s = \varphi_n(v)\}$ is FA-recognizable for every $s \in S$. Let $L = \{w \in L_n \mid \varphi_n(w) \in G\}$ and ψ be the restriction of φ_n onto L . Then $\psi : L \rightarrow G$ is a Cayley p.f. linear-time computable representation of G . \square

Corollary 2. *A virtually polycyclic group is Cayley p.f. linear-time computable.*

Proof. By Theorem 7, it is enough only to show that a polycyclic group is Cayley p.f. linear-time computable. Auslander showed that a polycyclic group has a faithful representation in $\text{SL}(n, \mathbb{Z})$ [18]. Therefore, a polycyclic group is isomorphic to a f.g. subgroup of $\text{GL}(n, \mathbb{Q})$ which is Cayley p.f. linear-time computable by Theorem 9. \square

4. Cayley Polynomial-Time Computable Groups

The notion of a Cayley p.f. linear-time computable group can be extended further to the notion of a Cayley polynomial-time computable group which

we introduce in this section.

We say that a function $f : \Sigma^* \rightarrow \Sigma^*$ is polynomial-time computable if it is computed by a deterministic one-tape Turing machine in time $O(p(n))$, where $p(n)$ is a polynomial and n is a length of the input. Note that (in contrast to the linear-time case) restricting to position-faithful Turing machines has no effect: a function computed by a deterministic one-tape Turing machine in polynomial time can be computed by a deterministic position-faithful one-tape Turing machine in polynomial time by performing the same steps and at the end copying the output to the front of the tape (this takes at most polynomial time).

Let G be a f.g. group and $S = \{s_1, \dots, s_n\} \subseteq G$ be a finite set of semigroup generators of G . Let \mathcal{C} be a nonempty class of languages.

Definition 4 (Cayley polynomial-time computable groups). *We say that the group G is \mathcal{C} -Cayley polynomial-time computable if there exist a language $L \subseteq \Sigma^*$ from the class \mathcal{C} over some finite alphabet Σ , a bijective mapping $\psi : L \rightarrow G$ between the language L and the group G and polynomial-time computable functions $f_i : \Sigma^* \rightarrow \Sigma^*$ such that $f_i(L) \subseteq L$ and for every $w \in L$: $\psi(f_i(w)) = \psi(w)s_i$, for all $i = 1, \dots, n$. We call $\psi : L \rightarrow G$ a \mathcal{C} -Cayley polynomial-time computable representation of the group G . If the requirement for L to be in a specific class \mathcal{C} is omitted, then we just say that G is a Cayley polynomial-time computable group and $\psi : L \rightarrow G$ is a Cayley polynomial-time computable representation of G .*

A \mathcal{C} -Cayley p.f. linear-time computable group is \mathcal{C} -Cayley polynomial-time computable. Similarly to \mathcal{C} -Cayley p.f. linear-time computable groups, the notion of a \mathcal{C} -Cayley polynomial-time computable group does not depend on the choice of generators.

Proposition 3. *The notion of a \mathcal{C} -Cayley polynomial-time computable group does not depend on the choice of generators.*

Proof. We first notice that if given functions $f_{j_i} : \Sigma^* \rightarrow \Sigma^*$, $i = 1, \dots, m$ are polynomial-time computable, then the composition $f_{j_m} \circ \dots \circ f_{j_1}$ is polynomial-time computable. The rest literally repeats the proof of Proposition 1 modulo changing the term automatic (p.f. linear-time) to polynomial-time. \square

Remark 8. *We note that if the degree of a polynomial $p(n)$ is greater than one, then the composition $g \circ f$ of two functions f and g computed in time*

$O(p(n))$ is in general not necessarily computed in time $O(p(n))$; we may only guarantee it is computed in time $O(p(p(n)))$. So, fixing an upper bound for the time complexity in Definition 4 one loses the independence on the choice generators. An alternative approach could be to use a more powerful computational model (for example, a two-tape Turing machine) and force the time complexity to be at most linear. In this case one gets the independence on the choice generators without necessity to update the time complexity.

Following [10, Definition 4] we introduce the notion of a Cayley polynomial-time computable representation which has quasigeodesic normal form (n.f.). Let A be some set of generators of G .

Definition 5. We say that a \mathcal{C} -Cayley polynomial-time computable representation $\psi : L \rightarrow G$ has quasigeodesic normal form if there is a constant C such that for all $w \in L$: $|w| \leq C(d_A(\psi(w)) + 1)$. In this case we say that G is \mathcal{C} -Cayley polynomial-time computable with quasigeodesic normal form. If the requirement for L to be in a specific class \mathcal{C} is omitted, then we just say that G is Cayley polynomial-time computable with quasigeodesic n.f. and a Cayley polynomial-time computable representation $\psi : L \rightarrow G$ has quasigeodesic n.f.

We note that a Cayley polynomial-time computable representation $\psi : L \rightarrow G$ does not necessary have quasigeodesic normal form like every Cayley p.f. linear-time computable representation (see Theorem 1). Therefore, the argument used in Theorem 2 cannot be generalized for an arbitrary Cayley-polynomial time computable representation. However, the following analogue of Theorem 2 holds:

Theorem 10 (Computing normal form in polynomial time). *Suppose that a Cayley polynomial-time computable representation $\psi : L \rightarrow G$ has quasigeodesic normal form. Then there is an algorithm which for a given input word $v = b_1 \dots b_k \in (A \cup A^{-1})^*$ computes the string $u \in L$ for which $\psi(u) = \pi(v)$. Moreover, this algorithm can be implemented by a deterministic one-tape Turing machine in polynomial time.*

Proof. The proof repeats Theorem 2 modulo the following minor changes. Since $\psi : L \rightarrow G$ has quasigeodesic normal form, for every string $u_{j-1} = \psi^{-1}(b_1 \dots b_{j-1})$, $j = 1, \dots, k$, the following inequality is satisfied: $|u_{j-1}| \leq C(d_A(b_1 \dots b_{j-1}) + 1) \leq C(j-1) + C \leq Ck + C$ for some constant C . Therefore, polynomial time is required to compute the string u_j from u_{j-1} . So the total

time required to compute the string u_k from the input $b_1 \dots b_k$ is polynomial. \square

Similarly to Corollary 1 we immediately obtain the following.

Corollary 3 (Solving word problem in polynomial time). *If a given group G is Cayley polynomial-time computable with quasigeodesic normal form, the word problem in G can be solved by a deterministic one-tape Turing machine in polynomial time.*

Clearly, the analogue of Theorem 3 holds for a Cayley polynomial-time computable representation $\psi : L \rightarrow G$.

Theorem 11. *For every Cayley polynomial-time computable representation $\psi : L \rightarrow G$ the language L is in the class RE.*

Furthermore, all closure properties with respect to taking a finite extension, the direct product, the free product and a finitely generated subgroup, shown in Theorems 4, 5, 6 and 7, respectively, remain valid for Cayley polynomial-time computable groups and the ones with quasigeodesic normal forms. Namely, we have the following.

Theorem 12 (Finite extensions, direct products, free products). *For a given class of languages \mathcal{C} , assuming that the relevant conditions are satisfied⁶, the class of \mathcal{C} -Cayley polynomial-time computable groups is closed under taking a finite extension, the direct product and the free product. The same holds for the class of \mathcal{C} -Cayley polynomial-time computable groups with quasigeodesic normal forms.*

Proof. For the first statement of the theorem the proof repeats Theorems 4, 5 and 6 with minor obvious changes. For the second statement of the theorem it is enough to notice that the quasigeodesic property is preserved for all representations which appear in the proofs of these theorems. \square

Theorem 13 (Finitely generated subgroups). *The class of Cayley polynomial-time computable groups is closed under taking a finitely generated subgroup. The same holds for the class of Cayley polynomial-time computable groups with quasigeodesic normal forms.*

⁶See the conditions on the class \mathcal{C} in Theorems 4, 5 and 6 for finite extensions, direct products and free products, respectively.

Proof. For the first statement of the theorem the proof repeats Theorem 7 with obvious changes. In order to show the second statement of the theorem, in the proof of Theorem 7 it is enough to notice that if $\psi : L \rightarrow G$ has quasigeodesic normal form, then for each $w \in L' \subset L$ the inequalities $|w| \leq C(d_A(\psi(w)) + 1) \leq C(C'd_{A'}(\psi(w)) + 1)$ hold for some constants $C, C' > 0$. This implies that $\psi' : L' \rightarrow H$, which is the restriction of ψ onto L' , also has quasigeodesic normal form. \square

We note that Theorem 8 cannot be directly generalized to \mathcal{C} -Cayley polynomial – time computable groups, so in general we cannot say how they compare with \mathcal{C} -graph automatic groups. However, for some special classes of languages \mathcal{C} , it is possible to relate these two classes of groups, see Remark 9 below.

Remark 9. *In [10, Theorem 10] the second author and Taback showed that for a \mathcal{S}_k -graph automatic group with quasigeodesic normal form, the normal form is computable in polynomial time, where \mathcal{S}_k is the class of languages accepted by a non-blind non-deterministic k -counter automaton running in quasi-realtime. We recall that such automaton is a non-deterministic automaton augmented with k integer counters which are initially set to zero [19]. These counters can be incremented, decremented, set to zero and compared to zero. Running in quasi-realtime means that the number of allowed consecutive ϵ -transitions is bounded from above by some constant. A string is accepted by this automaton exactly if it reaches an accepting state with all counters returned to zero.*

Now let G be a \mathcal{S}_k -graph automatic group with a symmetric set of semi-group generators $S = \{s_1, \dots, s_n\} \subset G$. Then there is a bijection $\psi : L \rightarrow G$ from a language $L \subseteq \Sigma^$ in the class \mathcal{S}_k to G for which the languages $L_s = \{u \otimes v \mid u, v \in L, \psi(v) = \psi(u)s\}$ are in \mathcal{S}_k for every $s \in S$. Assume that there is a polynomial p such that for all $s \in S$ and $u, v \in L$ for which $\psi(v) = \psi(u)s$ the inequality $|v| \leq p(|u|)$ holds; we note that the latter necessarily holds if $\psi : L \rightarrow G$ is a Cayley polynomial-time computable representation. Then G is \mathcal{S}_k -Cayley polynomial-time computable and $\psi : L \rightarrow G$ is a \mathcal{S}_k -Cayley polynomial-time computable representation of G . In order to prove this one only needs to show that for a given $s \in S$ there is a polynomial-time algorithm which for the input $u \in L$ produces the output $v \in L$ such that $u \otimes v \in L_s$. The reader may look up this algorithm and the explanation why it runs in polynomial time in [10, Theorem 10].*

What are examples of Cayley polynomial-time computable groups? Especially we are interested in examples of REG-Cayley polynomial-time computable groups because, similarly to Cayley p.f. linear-time computable groups, this class naturally extends the class of Cayley automatic groups.

First, in order to show that the class of REG-Cayley polynomial-time computable groups is wide, we notice that it comprises all f.g. nilpotent groups. Let G be a f.g. nilpotent group. Suppose first that G is torsion-free. There is a central series $G = G_1 \geq \dots \geq G_{n+1} = 1$ such that G_i/G_{i+1} is an infinite cyclic group for all $i = 1, \dots, n$. Then there exist $a_1, \dots, a_n \in G$ for which $G_i = \langle a_i, G_{i+1} \rangle$. This implies that every element $g \in G$ has a unique normal form $g = a_1^{x_1} \dots a_n^{x_n}$, where x_1, \dots, x_n are integers. Let L be a language of such normal forms over the alphabet $\Sigma = \{a_1, \dots, a_n, a_1^{-1}, \dots, a_n^{-1}\}$. Clearly, L is a regular language. The canonical mapping $\pi : L \rightarrow G$ gives a bijection between L and G . For given two group elements $g_1 = a_1^{x_1} \dots a_n^{x_n}$ and $g_2 = a_1^{y_1} \dots a_n^{y_n}$, the product $g_1 g_2$ equals $a_1^{q_1} \dots a_n^{q_n}$ for some integers q_1, \dots, q_n . This defines the functions $q_i(x_1, \dots, x_n, y_1, \dots, y_n)$, $i = 1, \dots, n$ of $2n$ integer variables $x_1, \dots, x_n, y_1, \dots, y_n$; in fact it can be shown that q_i depends only on x_1, \dots, x_i and y_1, \dots, y_i for every $i = 1, \dots, n$. Hall showed [20] that the functions q_i are polynomials $q_i \in \mathbb{Q}[x_1, \dots, x_n, y_1, \dots, y_n]$. Therefore, for each semigroup generator $s \in \{a_1, \dots, a_n, a_1^{-1}, \dots, a_n^{-1}\}$ there exist polynomials $p_{s,i} \in \mathbb{Q}[x_1, \dots, x_n]$ for $i = 1, \dots, n$ such that $a_1^{x_1} \dots a_n^{x_n} s = a_1^{p_{s,1}} \dots a_n^{p_{s,n}}$. It can be seen that these right multiplications are polynomial-time computable functions. Therefore, $\pi : L \rightarrow G$ is a REG-Cayley polynomial-time computable representation. If the group G is not torsion-free, it has a torsion-free nilpotent subgroup of finite index. Therefore, by Theorem 12, G also has a REG-Cayley polynomial-time computable representation.

Other nontrivial examples of REG-Cayley polynomial-time computable groups include the wreath product $\mathbb{Z}_2 \wr \mathbb{Z}^2$ and Thompson's group F . We denote by IND the class of indexed languages. In [21, S 5] the first author and Khoussainov showed that the group $\mathbb{Z}_2 \wr \mathbb{Z}^2$ is (REG, IND)-graph automatic⁷ by constructing a certain bijection between a regular language and the group $\mathbb{Z}_2 \wr \mathbb{Z}^2$. It can be verified that this bijection is a REG-Cayley polynomial-time computable representation of $\mathbb{Z}_2 \wr \mathbb{Z}^2$. Therefore, $\mathbb{Z}_2 \wr \mathbb{Z}^2$ is a REG-Cayley polynomial-time computable group. This representation does not

⁷ (\mathcal{B}, \mathcal{C})-graph automatic is defined in [10]: the normal form is in the language class \mathcal{B} and the 2-tape language for multiplication is in the class \mathcal{C} .

have quasigeodesic normal form, see [21, Remark 9].

Let \mathcal{DS}_1 be the class of non-blind deterministic 1-counter languages (see Remark 9 where the definition of non-blind non-deterministic k -counter languages is recalled). In [22] the second author and Taback showed that Thompson's group F is $(\text{REG}, \mathcal{DS}_1)$ -graph automatic. It follows from the metric inequalities established in [22, Proposition 3.3] and the observation made in the end of Remark 9 that F is REG-Cayley polynomial-time computable group with quasigeodesic normal form.

For the last example of a Cayley polynomial-time computable group we mention the wreath product $\mathbb{Z}_2 \wr \mathbb{F}_2$. We denote by DCFL the class of deterministic context-free languages. In [21, S 4] it was shown that the group $\mathbb{Z}_2 \wr \mathbb{F}_2$ is DCFL-graph automatic by constructing a certain bijection between a DCFL language and the group $\mathbb{Z}_2 \wr \mathbb{F}_2$. It can be verified that this bijection is a DCFL-Cayley polynomial-time computable representation of $\mathbb{Z}_2 \wr \mathbb{F}_2$. Moreover, it immediately follows from the metric inequalities shown in [21, Theorem 5] that this representation has quasigeodesic normal form. Therefore, $\mathbb{Z}_2 \wr \mathbb{F}_2$ is a DCFL-Cayley polynomial-time computable group with quasigeodesic normal form.

5. Cayley distance function for Cayley polynomial-time computable groups

Let G be a f.g. group with a finite generating set $A \subset G$. Let $\psi : L \rightarrow G$ be a bijection from a language $L \subseteq \Sigma^*$ to the group G . For each symbol $\sigma \in \Sigma$ one can assign a group element $g_\sigma \in G$. This assignment defines a mapping $\alpha : \Sigma \rightarrow G$, not necessarily injective, for which $\alpha(\sigma) = g_\sigma$ for all $\sigma \in \Sigma$. Then we can define the canonical mapping $\pi_\alpha : L \rightarrow G$ as follows: for a given string $w = \sigma_1 \dots \sigma_k \in L$ we define $\pi_\alpha(w) \in G$ as $\pi_\alpha(w) = \alpha(\sigma_1)\alpha(\sigma_2) \dots \alpha(\sigma_k)$ and $\pi_\alpha(w) = e$ if $w = \epsilon$. Thus, for fixed $\psi : L \rightarrow G$ and $\alpha : \Sigma \rightarrow G$, the following nondecreasing function $h_{\psi, \alpha} : [N, +\infty) \rightarrow \mathbb{R}^+$ is defined by:

$$h_{\psi, \alpha}(n) = \max\{d_A(\pi_\alpha(w), \psi(w)) \mid w \in L^{\leq n}\}, \quad (2)$$

where $d_A(\pi_\alpha(w), \psi(w))$ is the distance between $\pi_\alpha(w)$ and $\psi(w)$ in the word metric relative to A , $L^{\leq n} = \{w \in L \mid |w| \leq n\}$ and $N = \min\{n \in \mathbb{N} \mid L^{\leq n} \neq \emptyset\}$. For given ψ and α we call $h_{\psi, \alpha}$ a *Cayley distance function*. This function was introduced in [4] and studied in [5, 6] in the context of Cayley automatic

groups⁸.

Clearly, if G is automatic, then for an automatic representation $\pi : L \rightarrow G$, $L \subset (A \cup A^{-1})^*$ and a natural mapping $\alpha : A \cup A^{-1} \rightarrow G$ for which $\alpha(s) = s$, $s \in A \cup A^{-1}$ all values of the Cayley distance function $h_{\pi, \alpha}$ are equal to zero. [4, Theorem 8] shows that if a group G has some Cayley automatic representation $\psi : L \rightarrow G$ and mapping $\alpha : \Sigma \rightarrow G$ for which the Cayley distance function $h_{\psi, \alpha}$ is bounded from above by a constant, then G must be automatic. In [6] the first two authors and Taback ask: can the Cayley distance function can become arbitrarily close to a constant function for some non-automatic Cayley automatic group? Here we show that the answer is no when we generalise to Cayley p.f. linear-time computable and REG-Cayley polynomial-time computable representations: we furnish examples which have Cayley p.f. linear-time computable and REG-Cayley polynomial-time computable representations for which the Cayley distance function is zero.

For given two nondecreasing functions $h_1 : [N_1, +\infty) \rightarrow \mathbb{R}^+$ and $h_2 : [N_2, +\infty) \rightarrow \mathbb{R}^+$ we say that $h_1 \preceq h_2$, if there exist integer constants $K, M > 0$ and $N \geq \max\{N_1, N_2\}$ such that $h_1(n) \leq Kh_2(Mn)$ for all $n \geq N$. It is said that $h_1 \asymp h_2$ if $h_1 \preceq h_2$ and $h_2 \preceq h_1$. We say that a Cayley automatic group G is separated from automatic groups if there exists a non-decreasing unbounded function f such that $f \preceq h_{\psi, \alpha}$ for all Cayley automatic representations $\psi : L \rightarrow G$ and mappings $\alpha : \Sigma \rightarrow G$. We denote by $\mathbf{z} : \mathbb{N} \rightarrow \mathbb{R}^+$ the zero function: $\mathbf{z}(n) = 0$ for all $n \in \mathbb{N}$. We say that a Cayley distance function $h_{\psi, \alpha}$ vanishes if $h_{\psi, \alpha} \asymp \mathbf{z}$: this equivalently means that $h_{\psi, \alpha}(n) = 0$ for all $n \in \text{dom } h_{\psi, \alpha}$. We denote by $\mathbf{i} : \mathbb{N} \rightarrow \mathbb{R}^+$ the identity function: $\mathbf{i}(n) = n$ for all $n \in \mathbb{N}$.

Theorem 14. *There exists a Cayley automatic group G separated from automatic groups but for which the Cayley distance function $h_{\psi, \alpha}$ vanishes for some Cayley p.f. linear-time computable representation ψ of G and a mapping α .*

Proof. In order to prove the theorem one needs to provide an example of a group G satisfying the condition of the theorem. For such an example we

⁸In [4, 5] it is assumed from the beginning that L is a language over some symmetric set of generators. It can be seen that this assumption is purely a matter of convenience and it does not have any effect on the study of the Cayley distance function $h_{\psi, \alpha}$.

take the lamplighter group $G = \mathbb{Z}_2 \wr \mathbb{Z}$. The lamplighter group $\mathbb{Z}_2 \wr \mathbb{Z}$ is Cayley automatic [1]; but not automatic because it is not finitely presented [2]. By [4, Theorem 13], for every Cayley automatic representation $\psi : L \rightarrow \mathbb{Z}_2 \wr \mathbb{Z}$ and a mapping $\alpha : \Sigma \rightarrow \mathbb{Z}_2 \wr \mathbb{Z}$ the corresponding function $h_{\psi, \alpha}$ given by (2) is coarsely greater or equal than $\mathbf{i} : \mathbf{i} \preceq h_{\psi, \alpha}$. Below we will show that there exist a Cayley p.f. linear-time computable representation of $\mathbb{Z}_2 \wr \mathbb{Z}$ and a mapping for which the Cayley distance function vanishes.

Each element of the lamplighter group $\mathbb{Z}_2 \wr \mathbb{Z}$ is identified with a pair (f, z) , where f is a function $f : \mathbb{Z} \rightarrow \{0, 1\}$ with finite support (that is, only for finitely many integers i , $f(i) = 1$) and z is an integer. We denote by a the pair $(f_0, 1)$, where $f_0(j) = 0$ for all $j \in \mathbb{Z}$, and by b the pair $(f_1, 0)$, where $f_1(j) = 0$ for all $j \neq 0$ and $f_1(0) = 1$. The group elements a and b generate $\mathbb{Z}_2 \wr \mathbb{Z}$ and the right multiplications by a, a^{-1} and $b = b^{-1}$ are as follows. For a given $g = (f, z) \in \mathbb{Z}_2 \wr \mathbb{Z}$, $ga = (f, z + 1)$, $ga^{-1} = (f, z - 1)$ and $gb = gb^{-1} = (f', z)$, where $f'(i) = f(i)$ for $i \neq z$, $f'(z) = 0$ if $f(z) = 1$ and $f'(z) = 1$ if $f(z) = 0$. The identity e of $\mathbb{Z}_2 \wr \mathbb{Z}$ corresponds to the pair $(f_0, 0)$.

Let $g = (f, z)$ be a given group element of $\mathbb{Z}_2 \wr \mathbb{Z}$. Let $r = \max\{z, i \mid f(i) = 1, i \in \mathbb{Z}\}$ and $\ell = \min\{z, i \mid f(i) = 1, i \in \mathbb{Z}\}$. Let $\Sigma = \{b, a, a^{-1}, \uparrow, \#\}$. Let us define a normal form $w \in \Sigma^*$ of g to be the string $w = a^\ell \# u \# a^{z-r}$, where the string u is as follows. Initially $i = \ell$, the procedure to construct the string u is the following:

- If $f(i) = 1$, we write b as the first symbol of u . If $z = i$, we write \uparrow as the next symbol of u . Update $i \rightarrow i + 1$;
- If $i \leq r$, we write a as the the next symbol of u , then b if $f(i) = 1$ and then \uparrow if $z = i + 1$. Update $i \rightarrow i + 1$;
- If $i \leq r$, we write a as the next symbol of u , then b if $f(i) = 1$ and then \uparrow if $z = i + 2$. Update $i \rightarrow i + 1$;
- This process is continued until $i \leq r$. The result is the string u .

Informally speaking, the normal form $w = a^\ell \# u \# a^{z-r}$ is obtained as follows. First the pointer is moved from the position $i = 0$ to the position $i = \ell$. After that the pointer is moved to the right scanning the values $f(i)$ and the position of the lamplighter z until it reaches the position $i = r$. Then the pointer moves to the left until it reaches the position of the lamplighter $i = z$. Let us give two examples. Let $g_1 = (f, z)$ be the pair for which $z = 1$,

$f(-1) = 1, f(0) = 1, f(2) = 1$ and $f(i) = 0$ for all $i \neq -1, 0, 2$. The normal form of g_1 is $u = a^{-1}\#baba \uparrow ab\#a^{-1}$. Let $g_2 = (f, z)$ be the pair for which $z = 1, f(-2) = 1$ and $f(i) = 0$ for all $i \neq -2$. The normal form of g_2 is $u = a^{-1}a^{-1}\#baaa \uparrow \#$.

Let L be the language of all such normal forms. We denote by \mathcal{C}_1 the class of languages recognized by a (quasi–realtime) blind deterministic 1–counter automaton⁹. It follows from the simple argument below that the language L is in the class \mathcal{C}_1 . Let $w = a^\ell\#u\#a^{z-r}$. The string u is of the form $u = p \uparrow s$, where p is the prefix of u preceding the symbol \uparrow and s is the suffix. The counter is increased by one each time the automaton reads the symbol a in the suffix s of u . The counter is decreased by one each time the automaton reads the symbol a^{-1} in the suffix a^{z-r} of w . Then $w \in L$ if and only if the counter returns to 0.

Construction of automatic functions recognizing the right multiplications by a, a^{-1} and b is easy, so we skip it for brevity. Thus, we constructed a \mathcal{C}_1 –Cayley p.f. linear–time computable representation $\psi : L \rightarrow \mathbb{Z}_2 \wr \mathbb{Z}$ of the lamplighter group $\mathbb{Z}_2 \wr \mathbb{Z}$ which sends a normal form $w = a^\ell\#u\#a^{z-r}$ to the corresponding group element $g = (f, z)$. Now let $\alpha : \Sigma \rightarrow \mathbb{Z}_2 \wr \mathbb{Z}$ be the following mapping: $\alpha(a) = a, \alpha(a^{-1}) = a^{-1}, \alpha(b) = b, \alpha(\uparrow) = e$ and $\alpha(\#) = e$. Clearly, for the \mathcal{C}_1 –Cayley p.f. linear–time computable representation ψ and the mapping α , the Cayley distance function $h_{\psi, \alpha}$ vanishes. \square

Theorem 15. *There exist a Cayley automatic group G separated from automatic groups but for which the Cayley distance function $h_{\psi, \alpha}$ vanishes for some REG–Cayley polynomial–time computable representation ψ of G and a mapping α .*

Proof. Let us consider the Baumslag–Solitar groups $BS(p, q) = \langle a, t \mid ta^p t^{-1} = a^q \rangle$ for $1 \leq p < q$. These groups are not automatic [2], but they are Cayley automatic [1, 11]. By [5, Corollary 2.4], for every Cayley automatic representation $\psi : L \rightarrow BS(p, q)$ and a mapping $\alpha : \Sigma \rightarrow BS(p, q)$ the corresponding function $h_{\psi, \alpha}$ given by (2) is coarsely greater or equal than \mathbf{i} : $\mathbf{i} \preceq h_{\psi, \alpha}$. We will show that for the Baumslag–Solitar group $BS(p, q)$ there are a REG–Cayley

⁹We recall that a blind deterministic 1–counter automaton is a finite automaton augmented by an integer counter, initially set to zero, which can be incremented and decremented, but not read. A string is accepted by such an automaton exactly if it reaches an accepting state with the counter returned to zero.

polynomial-time computable representation and a mapping for which the Cayley distance function vanishes.

As a HNN extension of the infinite cyclic group the Baumslag–Solitar group $BS(p, q)$ admits the following normal form, see, e.g., [23, Chapter IV]. Every group element $g \in BS(p, q)$ can be uniquely written as a freely reduced word over the alphabet $\Sigma = \{a, a^{-1}, t, t^{-1}\}$ of the form $w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k$, where $\varepsilon_i \in \{+1, -1\}$, $k \in \mathbb{Z}$, $w_i = \{\varepsilon, a, \dots, a^{p-1}\}$ if $\varepsilon_i = -1$ and $w_i = \{\varepsilon, a, \dots, a^{q-1}\}$ if $\varepsilon_i = +1$. The language L of such normal forms is clearly regular. For a bijection between the language L and the group $BS(p, q)$ we take the canonical mapping: $\pi : L \rightarrow G$. The right multiplications by a and a^{-1} are as follows: $w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times a} w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^{k+1}$, $w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times a^{-1}} w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^{k-1}$. Let $k = mq + r$ for $m \in \mathbb{Z}$ and $r \in \{0, 1, \dots, q-1\}$. The right multiplication by t is as follows (different cases are considered separately):

- if $r \neq 0$, then $w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times t} w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^r t a^{mp}$;
- if $r = 0$, $\ell \geq 1$ and $\varepsilon_1 = +1$, then

$$w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times t} w_\ell t^{\varepsilon_\ell} \dots w_1 t t a^{mp};$$

- if $r = 0$, $\ell \geq 1$ and $\varepsilon_1 = -1$, then

$$w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times t} w_\ell t^{\varepsilon_\ell} \dots w_2 t^{\varepsilon_2} w_1 a^{mp};$$

- if $r = 0$ and $\ell = 0$, then $a^k \xrightarrow{\times t} t a^{mp}$.

Let $k = np + s$ for $n \in \mathbb{Z}$ and $s \in \{0, 1, \dots, p-1\}$. The right multiplication by t^{-1} is as follows:

- if $s \neq 0$, then $w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times t^{-1}} w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^s t^{-1} a^{nq}$;
- if $s = 0$, $\ell \geq 1$ and $\varepsilon_1 = +1$, then

$$w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times t^{-1}} w_\ell t^{\varepsilon_\ell} \dots w_2 t^{\varepsilon_2} w_1 a^{nq};$$

- if $s = 0$, $\ell \geq 1$ and $\varepsilon_1 = -1$, then

$$w_\ell t^{\varepsilon_\ell} \dots w_1 t^{\varepsilon_1} a^k \xrightarrow{\times t^{-1}} w_\ell t^{\varepsilon_\ell} \dots w_1 t^{-1} t^{-1} a^{nq};$$

- if $s = 0$ and $\ell = 0$, then $a^k \xrightarrow{\times t^{-1}} t^{-1}a^{nq}$.

It can be seen that each of the right multiplications by a, a^{-1}, t and t^{-1} shown above is polynomial-time computable. Therefore, $\pi : L \rightarrow BS(p, q)$ is a REG-Cayley polynomial-time computable representation. Moreover, for $\pi : L \rightarrow BS(p, q)$ and a natural mapping $\alpha : \Sigma \rightarrow BS(p, q)$, for which $\alpha(a) = a, \alpha(a^{-1}) = a^{-1}, \alpha(t) = t$ and $\alpha(t^{-1}) = t^{-1}$, the Cayley distance function $h_{\pi, \alpha}$ vanishes. \square

Remark 10. *It follows from the metric estimates for the Baumslag-Solitar group $BS(p, q)$ obtained by Burillo and the second author [24] that the REG-Cayley polynomial-time computable representation $\pi : L \rightarrow BS(p, q)$ from the proof of Theorem 15 does not have quasigeodesic normal form; see also a proof of the analogous fact in [10, p. 317].*

6. Conclusion

In this paper we introduced the notion of a \mathcal{C} -Cayley p.f. linear-time computable and a \mathcal{C} -Cayley polynomial-time computable group which extend the notion of a Cayley automatic group introduced by Kharlampovich, Khoussainov and Miasnikov. We proved some algorithmic and closure properties for these groups, and showed examples. We analysed behaviour of the Cayley distance function for Cayley p.f. linear-time computable and REG-Cayley polynomial-time computable representations. For future work we plan to focus on the classes of Cayley p.f. linear-time computable and REG-Cayley polynomial-time computable groups.

References

- [1] O. Kharlampovich, B. Khoussainov, A. Miasnikov, From automatic structures to automatic groups, *Groups, Geometry, and Dynamics* 8 (1) (2014) 157–198.
- [2] D. B. A. Epstein, J. W. Cannon, D. F. Holt, S. V. F. Levy, M. S. Paterson, W. P. Thurston, *Word Processing in Groups*, Jones and Barlett Publishers. Boston, MA, 1992.
- [3] J. Case, S. Jain, S. Seah, F. Stephan, Automatic functions, linear time and learning, *Logical Methods in Computer Science* 9 (3:19).

- [4] D. Berdinsky, P. Trakuldit, Measuring closeness between Cayley automatic groups and automatic groups, in: S. Klein, C. Martn-Vide, D. Shapira (Eds.), *Language and Automata Theory and Applications*, Vol. 10792, Springer International Publishing, 2018, pp. 245–257.
- [5] D. Berdinsky, P. Trakuldit, Towards quantitative classification of Cayley automatic groups, *East–West J. of Mathematics* 20 (2) (2018) 107–124.
- [6] D. Berdinsky, M. Elder, J. Taback, Separating automatic from Cayley automatic groups, to be submitted (2020).
- [7] M. R. Bridson, R. H. Gilman, Formal language theory and the geometry of 3–manifolds, *Commentarii Mathematici Helvetici* 71 (1) (1996) 525–555.
- [8] G. Baumslag, M. Shapiro, H. Short, Parallel poly–pushdown groups, *Journal of Pure and Applied Algebra* 140 (3) (1999) 209–227.
- [9] M. Brittenham, S. Hermiller, D. Holt, Algorithms and topology of Cayley graphs for groups, *Journal of Algebra* 415 (2014) 112–136.
- [10] M. Elder, J. Taback, C –graph automatic groups, *Journal of Algebra* 413 (2014) 289–319.
- [11] D. Berdinsky, B. Khoussainov, On automatic transitive graphs, in: A. Shur, M. Volkov (Eds.), *Developments in Language Theory 2014*, Vol. 8633 of *Lecture Notes in Computer Science*, Springer International Publishing, 2014, pp. 1–12.
- [12] S. Bérubé, T. Palnitkar, J. Taback, Higher rank lamplighter groups are graph automatic, *Journal of Algebra* 496 (2018) 315–343.
- [13] B. Khoussainov, A. Nerode, Automatic presentations of structures, in: D. Leivant (Ed.), *Logic and Computational Complexity*, Vol. 960 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 1995, pp. 367–392.
- [14] F. Stephan, Automatic structures – recent results and open questions, *Journal of Physics: Conference Series* 622 (012013).
- [15] K. A. Mikhailova, The occurrence problem for direct products of groups, *Mat. Sb. (N.S.)* 70(112) (2) (1966) 241–251.

- [16] S. Jain, B. Khoussainov, F. Stephan, Finitely Generated Semiautomatic Groups, *Computability* 7 (2–3) (2018) 273–287.
- [17] A. Nies, P. Semukhin, Finite automata presentable abelian groups, *Annals of Pure and Applied Logic* 161 (3) (2009) 458–467.
- [18] L. Auslander, On a problem of Philip Hall, *Annals of Mathematics* 86 (1) (1967) 112–116.
- [19] R. Book, S. Ginsburg, Multi-stack-counter languages, *Math. Systems Theory* 6 (1972) 37–48.
- [20] P. Hall, Notes of Lectures given at the Canadian Mathematical Congress, University of Alberta, 1957.
- [21] D. Berdinsky, B. Khoussainov, Cayley automatic representations of wreath products, *International Journal of Foundations of Computer Science* 27 (2) (2016) 147 – 159.
- [22] M. Elder, J. Taback, Thompson’s group F is 1-counter graph automatic, *Groups Complexity Cryptology* 8 (1) (2016) 21–33.
- [23] R. C. Lyndon, P. E. Schupp, *Combinatorial group theory*, Springer-Verlag Berlin Heidelberg New York, 1977.
- [24] J. Burillo, M. Elder, Metric properties of Baumslag–Solitar groups, *International Journal of Algebra and Computation* 25 (5) (2015) 799–811.