Finitely generated groups with automatic presentations

Graham P. Oliver and Richard M. Thomas

Department of Computer Science, University of Leicester,

Leicester LE1 7RH, UK.

gpo1@mcs.le.ac.uk rmt@mcs.le.ac.uk

Abstract

A structure is said to be computable if its domain can be represented by a set which is accepted by a Turing machine and if its relations can then be checked using Turing machines. Restricting the Turing machines in this definition to finite automata gives us a class of structures with a particularly simple computational structure; these structures are said to have *automatic presentations*. Given their nice algorithmic properties, these have been of interest in a wide variety of areas.

An area of particular interest has been the classification of automatic structures. One of the prime examples has been the class of groups. We give a complete characterization in the case of finitely generated groups and show that such a group has an automatic presentation if and only if it is virtually abelian.

1 Introduction

In this report we will be concerned with structures; we first explain what this means. A *structure* $\mathcal{A} = (A, R_1, \ldots R_n)$ consists of:

- a set A, called the *domain* (or *universe*) of $\mathcal{A}$;
- for each i with $1 \leqslant i \leqslant n$, there exists $r_i \geqslant 1$ such that R_i is a subset of A^{r_i}; r_i is called the *arity* of R_i.

For example, a group can be viewed as a structure $(G, \circ, e, {}^{-1})$, where $\circ$ has arity 3, e has arity 1, and ${}^{-1}$ has arity 2. A natural area of research is to consider which structures are computable. Taking the Turing machine as our computational paradigm, a computable structure is one for which there exist Turing machines which 'check' the relations in the structure. More formally, a structure $\mathcal{A} = (A, R_1, \ldots R_n)$ is said to be *computable* if:

- the domain A of $\mathcal{A}$ is recognized by a Turing machine;
- for each relation R_i in $\mathcal{A}$, there is a decision-making Turing machine that, on input $(a_1, \ldots, a_{r_i})$, outputs true if $(a_1, \ldots, a_{r_i}) \in R_i$ and false otherwise.

When we say that A is recognized by a Turing machine, we mean that there is a set of symbols I such that A is a recursively enumerable subset of I^*. In fact, when we consider automatic presentations (see Section 2), we allow a mapping from a subset of I^* onto A; in this case we will also need an automaton to check when two words in I^* represent the same element of A. In general, the way that elements of A are represented in I^* is clearly important.

Examples of computable structures include rational vector spaces and free groups (as well as many others). A restriction of this idea has also been introduced, that of *p-time structures*. These are computable structures for which the time complexity of each of the associated Turing machines is polynomial; as an example take any recursive Boolean algebra. For more information see [4] for example.

Khoussainov and Nerode have introduced [11] a very interesting restriction of this general idea, to *automatic structures*, i.e. those structures whose domain and relations can be checked by finite automata as opposed to Turing machines. A structure isomorphic to an automatic structure is said to have an *automatic presentation*. Given their nice algorithmic properties and the diversity of natural examples of such structures, these have been of interest in a variety of research areas.

The general idea of using finite automata to read structures is not entirely new; for example, in group theory, a group is said to be *automatic* if, when we code elements of the group as strings of generators, there is a regular subset L of the set of all strings of generators such that there are finite automata to check multiplication of words in L by generators. This concept was introduced in [6], motivated by work in hyperbolic manifolds as well as a general interest in computing on groups. The considerable success of the theory of automatic groups gives one motivation to have a general notion of automatic structures.

However, there are other motivations for a general study of automatic structures, most importantly, perhaps, the decidability properties that come with finite automata. In particular, the first-order theory of an automatic structure is decidable. Another motivation for the study of automatic presentations is that of extending some of the techniques of finite model theory to infinite structures that have finite presentations; see [2, 3] for example.

One interesting result presented in [11] is that all finitely generated abelian groups have automatic presentations. Some natural and important questions follow from this:

- How far can this be extended - are there other groups with automatic presentations?
- What are the necessary or sufficient conditions for a group to have an automatic presentation?
- How does the class of groups with automatic presentations compare with that of automatic groups?

These questions are the main incentive for the work presented here. We note that, whilst an automatic group is finitely generated (this is essentially part of the definition) a group with an automatic presentation need not be; however, we will only be concerned with finitely generated groups in this paper and we give a complete answer to these questions in that case (see Theorems 6.3 and 7.1 below). In particular, we show that a finitely generated group has an automatic presentation if and only if it is virtually abelian, and hence that a finitely generated group with an automatic presentation is necessarily automatic (but the converse is false).

2 Automatic Presentations

Intuitively, a structure has an automatic presentation if finite automata can be constructed that read its domain and each of its relations, that is, a structure has an automatic presentation if its domain can be represented by a regular language such that each of its relations is also a regular language.

Before we present the formal definition, we need to introduce the idea of a "convolution" (see, for example, [11]). Finite automata read words over an alphabet; we need a systematic way of using finite automata to read tuples of words of different lengths. Suppose, for example, we have the triple (`bronze`, `silver`, `gold`); how are we going to make this into a form which can be read by a finite automaton? It would seem somewhat unbalanced to represent these linearly (say `bronze_silver_gold`), and, in fact, [2] shows this would be overly restrictive. A better option is to arrange things so that the n^{th} letters of each word are read together (for each n). We begin by aligning the letters vertically, filling the gaps in length with a *padding symbol* not in the original alphabet; we use $\square$ here for this new symbol. In our example, we have:

$$\begin{array}{cccccc} \mathtt{b} & \mathtt{r} & \mathtt{o} & \mathtt{n} & \mathtt{z} & \mathtt{e} \\ \mathtt{s} & \mathtt{i} & \mathtt{l} & \mathtt{v} & \mathtt{e} & \mathtt{r} \\ \mathtt{g} & \mathtt{o} & \mathtt{l} & \mathtt{d} & \square & \square \end{array}$$

We then associate together the n^{th} letters of the words, that is, we make new triples of the first, second, third letters, and so on. Our example then becomes:

$$(\mathtt{b},\ \mathtt{s},\ \mathtt{g})(\mathtt{r},\ \mathtt{i},\ \mathtt{o})(\mathtt{o},\ \mathtt{l},\ \mathtt{l})(\mathtt{n},\ \mathtt{v},\ \mathtt{d})(\mathtt{z},\ \mathtt{e},\ \square)(\mathtt{e},\ \mathtt{r},\ \square). \tag{1}$$

We now form the new alphabet

$$E^3_\square = ((E \cup \{\square\}) \times (E \cup \{\square\}) \times (E \cup \{\square\})) \setminus \{(\square, \square, \square)\}.$$

Clearly the new form of the example in (1) is a word over the alphabet $E^3_\square$. We shall say that a language of words such as (`bronze`, `silver`, `gold`) is accepted by a finite automaton if the set of new forms of the words is accepted by a finite automaton over $E^3_\square$.

Let I be an arbitary alphabet. Formally, we define the *convolution* of $(x_1,\ x_2, \ldots,\ x_n) \in (I^*)^n$, where $x_i = x_i^1 x_i^2 \ldots x_i^{p_i}$ $(x_i^j \in I)$, to be

$$\operatorname{conv}(x_1, \ldots,\ x_n) = (\bar{x}_1^1,\ \bar{x}_2^1, \ldots,\ \bar{x}_n^1)(\bar{x}_1^2,\ \bar{x}_2^2, \ldots,\ \bar{x}_n^2) \ldots (\bar{x}_1^p,\ \bar{x}_2^p, \ldots,\ \bar{x}_n^p),$$

where $p = \max\{p_i : 1 \leqslant i \leqslant n\}$ and, for some $\square \notin I$,

$$\bar{x}_i^j = \begin{cases} x_i^j & 1 \leqslant j \leqslant p_i \\ \square & p_i < j \leqslant p \end{cases}.$$

We now have our definition of an automatic presentation:

Definition 2.1. *A structure* $\mathcal{A} = (A, R_1, \ldots R_n)$ *has an automatic presentation (over an alphabet* I*) if*

1. *there is a language* L *over* I *and a map* $c : L \to A$ *such that* c *is surjective;*

2. L *is accepted by a finite automaton over* I*;*

3. $L_= = \{conv(x,\ y) : c(x) = c(y),\ x, y \in L\}$ *is accepted by a finite automaton over* $I_\square^2$*;*

4. *for each relation* R_i *in* $\mathcal{A}$*, the language*

$$L_{R_i} = \{conv(x_1, \ldots, x_{r_i}) : (c(x_1), \ldots, c(x_{r_i})) \in R_i\}$$

   is accepted by a finite automaton over $I_\square^{r_i}$*.*

The tuple $(I,\ L,\ c,\ L_=,\ (L_{R_i})_{1 \leqslant i \leqslant n})$ is called an *automatic presentation* for $\mathcal{A}$. The presentation is called *injective* if c is injective, and *binary* if $|I| = 2$. At one extreme, it is clear that all structures with a finite domain have an automatic presentation (as finite languages are regular); at the other, we see that the domain of such a structure must be countable. These facts will be implicitly assumed throughout.

An easy example of a structure with an automatic presentation is the semigroup $(\mathbb{N}, +)$. The elements of $\mathbb{N}$ are coded over $\{0, 1\}$ using their binary notation, but in reverse order; we can allow extra 0's for ease of exposition (but see part 2 of Proposition 3.1 below). The standard method of adding binary numbers then gives an automaton for checking addition in the semigroup. The machine has two main states; it is in one state when nothing is being carried in the calculation and the other state when 1 is being carried.

Other examples include:

- $(\mathbb{Q},\ \leqslant)$, the rational numbers under their natural order [11];
- ω^n $(n \in \mathbb{N})$ [11] (see also [5]);
- structures with only unary relations [11];
- vector spaces over finite fields [11].

3 Properties

In this section we list some properties of structures with automatic presentations. We start with two useful facts from [2] and [11]:

Proposition 3.1. *Let $\mathcal{A}$ be a structure with an automatic presentation; then:*

1. *$\mathcal{A}$ has a binary automatic presentation.*
2. *$\mathcal{A}$ has an injective automatic presentation.*

Remark 3.2. The proof that, if we have an automatic presentation

$$(I,\ L,\ c,\ L_{=},\ (L_{R_i})_{1 \leqslant i \leqslant n}),$$

then we have an injective automatic presentation $(J,\ K,\ c',\ K_{=},\ (K_{R_i})_{1 \leqslant i \leqslant n})$, uses the same alphabet and constructs a subset K of L; so we may put the two parts of Proposition 3.1 together and say that every structure with an automatic presentation has an injective binary automatic presentation. $\square$

A common (and useful) way of working with finite automata is to use predicate calculus. This arises from the closure of regular languages under (finite) union and intersection, complementation, and the definability of the existential and universal quantifiers. As such, if R_1 and R_2 are relations recognised by finite automata, then $R_1 \wedge R_2$, $R_1 \vee R_2$, $\neg R_1$, $\exists x(R_1)$ and $\forall x(R_1)$ are also all recognised by finite automata; see [6] for details. The proofs are all constructive, which gives the following results from [11]:

Theorem 3.3. *Let $\mathcal{A}$ be a structure with an automatic presentation; then:*

1. *if P is a first-order definable relation on $\mathcal{A}$ then P is decidable;*
2. *the first-order theory of $\mathcal{A}$ is decidable.*

Remark 3.4. For future reference we note that the point about the proof of Theorem 3.3 is that, if $\mathcal{A} = (A, R_1, \ldots R_n)$ is a structure with an automatic presentation, then any relation S built from the R_i using first order constructs is also recognizable by a finite automaton; as a result, we would also have an automatic presentation for the structure $\mathcal{B} = (A, R_1, \ldots R_n, S)$. $\square$

Theorem 3.3 has been extended further. Let $\exists^\infty$ be the quantifier "there exist infinitely many" and let FO denote first-order logic; we then have [2, 3]:

Theorem 3.5. *Let $\mathcal{A}$ be a structure with an automatic presentation; then the FO($\exists^\infty$) theory of $\mathcal{A}$ is decidable.*

For more information about decidability results (and model-theoretic results in general) see [2].

We now come to "interpretations". Let $\mathcal{A}$ and $\mathcal{B}$ be structures; then an (n-dimensional) *interpretation* I of $\mathcal{B}$ in $\mathcal{A}$ consists of:

- a formula $\delta_I(x_1, \ldots, x_n)$ in $\mathcal{A}$; this is called the *domain formula* of I;
- for each unnested atomic formula $\phi(y_1, \ldots, y_m)$ of $\mathcal{B}$, a formula

  $$\phi_I(\bar{x}_1, \ldots, \bar{x}_m)$$

  of $\mathcal{A}$, where the $\bar{x}_i$ are disjoint n-tuples of distinct variables; these are called the *defining formulae* of I;
- a surjective map $f_I : \delta_I(A^n) \to B$; this is called the *coordinate map* of I.

In addition we insist that, for all unnested atomic formulae of $\mathcal{A}$ and all $\bar{a}_i \in \delta_I(A^n)$, we have:

$$B \models \phi(f_I(\bar{a}_1), \ldots, f_I(\bar{a}_m)) \;\Leftrightarrow\; A \models \phi_I(\bar{a}_1, \ldots, \bar{a}_m).$$

If there is an interpretation of $\mathcal{B}$ in $\mathcal{A}$ then we will say that $\mathcal{B}$ is *interpretable* in $\mathcal{A}$ and that $\mathcal{B}$ is a *subinterpretation* of $\mathcal{A}$. If we fix a particular logic L, then we get L-interpretations. See [10] for more details.

The following model-theoretic result [2, 3], as well as being useful in its own right, has many interesting consequences:

Proposition 3.6. *Let $\mathcal{A}$ be a structure and $\mathcal{B}$ be a structure with an automatic presentation; if $\mathcal{A}$ is FO($\exists^\omega$)-interpretable in $\mathcal{B}$, then $\mathcal{A}$ has an automatic presentation.*

Finally we note that it is demonstrated in [2] that a structure has an automatic presentation if and only if it is interpretable in the structure

$$M(\Sigma) = (\Sigma^*, \preceq, (R_a)_{a\in\Sigma}, \mathrm{el})$$

where:

- $R_a(x, y)$ if $xa = y$;
- $x \preceq y$ if x is a prefix of y;
- $\mathrm{el}(x, y)$ if $|x| = |y|$.

4 Results on Groups

Before presenting our results on groups with automatic presentations, it is worth commenting on the form of the structure for groups.

We asserted above that groups could be considered as having structure $(G, \circ,\ e,\ ^{-1})$. However, it is common to view groups just as having a single binary operation, and so we would have a structure $(G, \circ)$. Which is correct?

In a sense, the answer depends on how you are considering the structures. As noted in [10], the main difference is that of substructures: the substructures of groups as structures $(G, \circ)$ need only be subsemigroups, whereas, with $(G, \circ,\ e,\ ^{-1})$, they must be subgroups. For our purposes, we needn't be too worried by this distinction. It is clear that, for the structure $(G, \circ)$, the properties of having an identity and having inverses are both first-order definable; so, if a group as a structure $(G, \circ)$ has an automatic presentation, then (as in Remark 3.4) this same presentation may be expanded to one for the structure $(G, \circ,\ e,\ ^{-1})$. With this in mind, we need only concentrate on $(G, \circ)$ in what follows.

The following result from [11] sums up much of what is already known concerning finitely generated groups with automatic presentations:

Proposition 4.1. *All finitely generated abelian groups have automatic presentations.*

We now need another definition. Let χ be a group property (such as being abelian); then a group G is said to be *virtually* (or *almost*) χ if G contains a subgroup of finite index with the property χ. We then have:

Theorem 4.2. *All finitely generated virtually abelian groups have an automatic presentation.*

Proof. Let G be a finitely generated group with an abelian subgroup A of finite index; then G is FO-interpretable in A (see [1] for example). The result follows from Propositions 3.6 and 4.1. □

Remark 4.3. We make a note before continuing. Suppose that G is a finitely generated virtually abelian group, so that G has an abelian subgroup A of finite index. Then A is finitely generated and hence is a direct product $C_1 \times C_2 \times \ldots C_k$ of cyclic groups. If we consider the subgroup B of A generated by the infinite groups C_i (i.e. ignore the C_i which are finite cyclic groups), then B has finite index in A, and hence has finite index in G.

Now B is a free abelian group isomorphic to $\mathbb{Z}^n = \mathbb{Z} \times \mathbb{Z} \times \ldots \times \mathbb{Z}$ for some n; so every finitely generated virtually abelian group has a free abelian subgroup of finite index. Moreover, if H is a subgroup of finite index in a group G, then there is a normal subgroup N of G contained in H with N also of finite index in G; as a subgroup of a free abelian group is free abelian, we have that every finitely generated virtually abelian group has a normal free abelian subgroup of finite index. □

Remark 4.4. It is possible to prove Theorem 4.2 from first principles by constructing appropriate automata. We give an outline of the proof here.

Let G be a finitely generated virtually abelian group. As in Remark 4.3, let $A = \langle x_1, x_2, \ldots, x_n \rangle$ be a normal subgroup of G of finite index isomorphic to $\mathbb{Z}^n$ and then let $T = \{t_1, t_2, \ldots, t_k\}$ be a set of coset representatives for A in G. The coding of the elements of G is fairly straightforward: we have a symbol t_j for the coset representative, a symbol for an n-tuple of +'s and −'s, and then symbols for n-tuples of 1's and 0's.

The point is that any element g of G can be expressed in the form $t_j a$ with $a \in A$, and then a can be written in the form $x_1^{\epsilon_1 m_1} x_2^{\epsilon_2 m_2} \ldots x_n^{\epsilon_n m_n}$ with $\epsilon_i \in \{1, -1\}$ and $m_i \in \mathbb{N}$ (if $m_i = 0$ we take $\epsilon_i = 1$). We then represent g as $t_j\,(\epsilon_1, \epsilon_2, \ldots, \epsilon_n)\,\mathrm{conv}(\overline{m_1}, \overline{m_2}, \ldots, \overline{m_n})$, where $\overline{m_i}$ is the representation of m_i in reverse order binary notation. Since A is normal in G, each $x_i t_j$ is of the form $t_j x_1^{u_{1,i,j}} x_2^{u_{2,i,j}} \ldots x_n^{u_{n,i,j}}$ for some $u_{h,i,j} \in \mathbb{Z}$; so multiplication in G is given by

$$\begin{aligned} t_i x_1^{a_1} \ldots x_n^{a_n} . t_j x_1^{b_1} \ldots x_n^{b_n} &= t_i t_j x_1^{a'_1} \ldots x_n^{a'_n} x_1^{b_1} \ldots x_n^{b_n} \\ &= t_i t_j x_1^{a'_1 + b_1} \ldots x_n^{a'_n + b_n} \end{aligned}$$

where

$$a'_i = \sum_{k=1}^{n} a_k u_{i,k,j}.$$

Now let t_k and $c_1, c_2, \ldots, c_n$ be such that $t_i t_j = t_k x_1^{c_1} \ldots x_n^{c_n}$; then

$$\begin{aligned} t_i t_j x_1^{a'_1 + b_1} \ldots x_n^{a'_n + b_n} &= (t_k x_1^{c_1} \ldots x_n^{c_n}) x_1^{a'_1 + b_1} \ldots x_n^{a'_n + b_n} \\ &= t_k x_1^{a'_1 + b_1 + c_1} \ldots x_n^{a'_n + b_n + c_n} \end{aligned}$$

Given all this, we first create different transitions in our automaton for each possible pair of t_i's, and then, from these different transitions, for each possible combination of + and −s. Then, based on the binary addition of n-tuples and taking into account the $u_{1,i,j}$ and c_i, we construct the rest of the automaton. The states, roughly, represent the current value of the carry in the addition. As the total amount carried at each stage is bounded by $n - 1$ we have a finite automaton. □

5 Growth

Currently there are not many techniques for showing that a structure does not have an automatic presentation. One such method follows from decidability: if a structure has undecidable $\mathrm{FO}(\exists^\omega)$ theory then it doesn't have an automatic presentation; for further conditions see [12, 14, 17] for example.

Another important method involves "growth". Let $\mathcal{A}$ be a structure with domain A and an injective automatic presentation, and fix one such presentation; then, for $x \in A$, let $l(x)$ denote the length of the coding of x in this presentation. We have the following result from [2]:

Theorem 5.1. *Let $f : A^n \to A$ be a first-order definable function on $\mathcal{A}$; then there exists a constant N such that*

$$\forall \overline{a} \in A^n,\ l(f(\overline{a})) \leqslant max\{l(a_0), \ldots, l(a_{n-1})\} + N.$$

In particular, this result has the following consequence for groups:

Corollary 5.2. *Let G be a group with an injective automatic presentation; then there is a constant N such that*

$$g_0 g_1 = g_2 \ \Rightarrow\ l(g_2) \leqslant max\{l(g_0),\, l(g_1)\} + N$$

for all $g_0, g_1, g_2 \in G$.

There is a corresponding notion of growth in group theory. Let G be a group with a finite generating set Δ, and assume that Δ is closed under taking inverses. Now let $\delta(g)$ be the minimum $n \in \mathbb{N}$ such that

$$g = a_1 a_2 \ldots a_n,\ \ a_i \in \Delta.$$

The *growth function* of G is then defined to be

$$\gamma(n) = |\{g \in G : \delta(g) \leqslant n\}|.$$

The nature of this function (in the sense of its being bounded above by a polynomial function, below by an exponential function, or neither of these), is independent of which particular finite generating set we choose. As such, the nature of the growth function (in this sense) is a property solely of the group (as opposed to the group together with a generating set). In the three cases we have mentioned, the group is said to have (respectively) *polynomial growth*, *exponential growth* or *intermediate growth*; see [8] for a survey on growth in groups. We now prove the following result:

Theorem 5.3. *If a group G has an automatic presentation then G has polynomial growth.*

Before we do this, we first prove a useful proposition:

Proposition 5.4. *With notation as above, let $R = max\{l(a) : a \in \Delta\}$; then there is a constant N such that, for all $m \geqslant 1$, we have*

$$max\{l(a_1 \dots a_m) : a_i \in \Delta\} \leqslant R + \lceil \log_2 m \rceil N.$$

Proof. Let N be the constant of Corollary 5.2. We proceed by induction on m.

We first consider the case $m = 1$. Here we clearly have

$$\max\{l(a_1) : a_1 \in \Delta\} = R = R + \lceil \log_2 1 \rceil N.$$

Now assume the result holds for $1 \leqslant m \leqslant k$. We split our consideration into two cases.

Case one: k is odd, say $k = 2r - 1$. Then, using Corollary 5.2, we have

$$\begin{aligned}
\max\{l(a_1 \dots a_{k+1}) : a_i \in \Delta\} &= \max\{l(a_1 \dots a_{2r}) : a_i \in \Delta\} \\
&\leqslant \max\{l(a_1 \dots a_r),\ l(a_{r+1} \dots a_{2r}) : a_i \in \Delta\} + N \\
&\leqslant \max\{R + \lceil \log_2 r \rceil N,\ R + \lceil \log_2 r \rceil N\} + N \\
&= R + \lceil \log_2 r \rceil N + N \\
&= R + (\lceil \log_2 r + 1 \rceil) N \\
&= R + \lceil \log_2 r + \log_2 2 \rceil N \\
&= R + \lceil \log_2 2r \rceil N \\
&= R + \lceil \log_2 (k+1) \rceil N
\end{aligned}$$

as required.

Case two: k is even, say $k = 2r$. This time we have

$$\begin{aligned}
\max\{l(a_1 \dots a_{k+1}) : a_i \in \Delta\} &= \max\{l(a_1 \dots a_{2r+1}) : a_i \in \Delta\} \\
&\leqslant \max\{l(a_1 \dots a_r),\ l(a_{r+1} \dots a_{2r+1}) : a_i \in \Delta\} + N \\
&\leqslant \max\{R + \lceil \log_2 r \rceil N,\ R + \lceil \log_2 (r+1) \rceil N\} + N \\
&= R + \lceil \log_2 (r+1) \rceil N + N.
\end{aligned}$$

Now, we can't proceed quite as easily as before, as we will only reach $k + 2$; we split our consideration of this cases into two subcases.

Subcase one: r is not of the form 2^x with $x \geqslant 1$.

The function $\lceil \log_2 y \rceil$ on $\{y \in \mathbb{N} : y > 0\}$ takes the same value on y and $y+1$ except when y is of the form 2^x; so, if $r \neq 2^x$, then $\lceil \log_2(r+1) \rceil = \lceil \log_2 r \rceil$. This gives

$$\begin{aligned} R + \lceil \log_2(r+1) \rceil N + N &= R + \lceil \log_2 r \rceil N + N \\ &= R + \lceil \log_2 r + 1 \rceil N \\ &= R + \lceil \log_2 r + \log_2 2 \rceil N \\ &= R + \lceil \log_2 2r \rceil N \\ &= R + \lceil \log_2(2r+1) \rceil N \\ &= R + \lceil \log_2(k+1) \rceil N. \end{aligned}$$

Subcase two: $r = 2^x$ $(x \geqslant 1)$.

Note first that

$$\begin{aligned} \lceil \log_2(k+1) \rceil &= \lceil \log_2(2r+1) \rceil \\ &= \lceil \log_2(2.2^x + 1) \rceil \\ &= \lceil \log_2(2^{x+1} + 1) \rceil \\ &= x + 2 \end{aligned}$$

Now

$$\begin{aligned} R + \lceil \log_2(r+1) \rceil N + N &= R + \lceil \log_2(r+1) + 1 \rceil N \\ &= R + \lceil \log_2(r+1) + \log_2 2 \rceil N \\ &= R + \lceil \log_2 2(r+1) \rceil N \\ &= R + \lceil \log_2 2(2^x+1) \rceil N \\ &= R + \lceil \log_2(2^{x+1}+2) \rceil N \\ &= R + (x+2)N \\ &= R + \lceil \log_2(k+1) \rceil N \end{aligned}$$

as required. □

Given Proposition 5.4, we can now prove Theorem 5.3:

Proof. By Remark 3.2 we may assume that the presentation for G is injective and binary. Then, as

$$\max\{l(a_1 \ldots a_m) : a_i \in \Delta\} \leqslant R + \lceil \log_2 m \rceil N$$

by Proposition 5.4, the number of possible codes for words of the form $a_1 \dots a_m$ is

$$\begin{aligned} 2^{R+\lceil \log_2 m \rceil N} &= 2^R (2^{\lceil \log_2 m \rceil})^N \\ &\leqslant 2^R (2^{\log_2 m + 1})^N \\ &= 2^R 2^N (2^{\log_2 m})^N \\ &= km^N, \text{ where } k = 2^R 2^N \text{ is a constant.} \end{aligned}$$

So we have at most km^N possible elements g in G with $\delta(g) = m$; as a result, we have

$$\begin{aligned} \gamma(n) &= |\{g \in G : \delta(g) \leqslant n\}| \\ &\leqslant k.1^N + k.2^N + \ldots + k.n^N \\ &\leqslant k.n^{N+1}. \end{aligned}$$

So G has polynomial growth as required. □

6 Classification

We now quote two substantial known theorems which enable us to give a complete classification as to which finitely generated groups have an automatic presentation (to some extent solving a problem of [13]). We first need some more definitions.

If G is a group and if H and K are subsets of G, then we let $[H, K]$ denote the set of all elements of G of the form $h^{-1}k^{-1}hk$ with $h \in H$ and $k \in K$. If H and K are subgroups of G, then $[H, K]$ is a subgroup of G and if, in addition, H and K are normal in G, then $[H, K]$ is a normal subgroup of G. We now define the following chains of normal subgroups of G:

$$\begin{array}{lll} G^{(0)} = G; & G^{(1)} = [G, G]; & G^{(2)} = [G^{(1)}, G^{(1)}]; \\ G^{(3)} = [G^{(2)}, G^{(2)}]; & \ldots\ldots & \end{array}$$

$$\begin{array}{lll} \gamma_0(G) = G; & \gamma_1(G) = [\gamma_0(G), G]; & \gamma_2(G) = [\gamma_1(G), G]; \\ \gamma_3(G) = [\gamma_2(G), G]; & \ldots\ldots & \end{array}$$

Note that $G \geqslant G^{(1)} \geqslant G^{(2)} \geqslant \ldots$ and that $G \geqslant \gamma_1(G) \geqslant \gamma_2(G) \geqslant \ldots$. A group G is said to be *solvable* if $G^{(r)} = \{e\}$ for some $r \in \mathbb{N}$ and *nilpotent* if $\gamma_r(G) = \{e\}$ for some $r \in \mathbb{N}$; in the first case we call r the *derived length* of G and, in the second case, r is called the *nilpotency class* of G. Any nilpotent group is necessarily solvable but the converse is false.

Given this, we can now state Gromov's classification [9] of groups with polynomial growth:

Theorem 6.1. *If a finitely generated group has polynomial growth then it is virtually nilpotent.*

Eršov showed [7] that a nilpotent group has decidable first order theory if and only if it is virtually abelian. This was generalized by Romanovskii [16] to virtually polyclic groups and then by Noskov [15], who showed that a virtually solvable group has decidable first order theory if and only if it is virtually abelian. The fact we need here is the following intermediate result:

Theorem 6.2. *Let G be a finitely generated virtually nilpotent group with decidable first order theory; then G is virtually abelian.*

These two results enable us to prove:

Theorem 6.3 (Classification). *Let G be a finitely generated group; then G has an automatic presentation if and only if G is virtually abelian.*

Proof. Assume that G has an automatic presentation. By Theorem 5.3, G has polynomial growth, and so, by Theorem 6.1, G is virtually nilpotent. By Theorem 3.3, G has decidable first-order theory, and so, by Theorem 6.2, G is virtually abelian.

The converse follows from Theorem 4.2. □

7 Automatic Groups

The theory of automatic groups (see [6] for example) was mentioned in the introduction as one of the motivations for studying structures with automatic presentations. Naturally the connections between the two notions have been remarked upon elsewhere; see [3] for example. We make some further comments on the relationship between these concepts here.

Given a finitely generated group $(G, \circ)$ with generators $a_1, \ldots, a_n$, we form a new structure $(G, R_1, \ldots, R_n)$ where $R_i(g, h)$ if and only if $g \circ a_i = h$; this new structure is called the *Cayley graph* of G.

It is remarked in [11] that [6], in effect, proves that, if G is a finitely generated abelian group, then the Cayley graph of G has an automatic presentation. Proposition 4.1 is somewhat stronger, taking (as it does) full multiplication rather than just multiplication by generators.

The results described earlier bear this out, although it is not as clear cut as this. In fact, [6] proves more than the fact that the Cayley graph of G has an automatic presentation: it proves that it has an automatic presentation with a predetermined language. The point is that, in [6], we don't have the freedom to choose any appropriate coding but we must use a set of semigroup generators for G (i.e. a subset of G that generates G as a semigroup) and then represent the elements of G as words in these generators.

This distinction is significant. Let H be the *Heisenberg group*, i.e. the group of matrices

$$\left\{ \begin{pmatrix} 1 & x & z \\ 0 & 1 & y \\ 0 & 0 & 1 \end{pmatrix} : x, y, z \in \mathbb{Z} \right\}.$$

It is noted in [3] that the Cayley graph of H has an automatic presentation, but that it is not an automatic group. As H is not virtually abelian, it also does not have an automatic presentation (as a group) by Theorem 6.3.

Considering only finitely generated groups, let `AutoPres` represent the class of groups with automatic presentations, `Automatic` represent the class of automatic groups, and `CayleyAutoPres` represent the class of groups whose Cayley graphs have automatic presentations. We have

Theorem 7.1. $\texttt{AutoPres} \subsetneq \texttt{Automatic} \subsetneq \texttt{CayleyAutoPres}$.

Proof. All virtually abelian groups are automatic, but there are plenty of groups (such as free groups) that are automatic but do not have automatic presentations; this gives the first (proper) inclusion. The automata required for automatic groups give automatic presentations for the Cayley graphs of these groups; however, the Cayley graph of the Heisenberg group has an automatic presentation, but the Heisenberg group is not automatic. This gives the second (proper) inclusion. □

Acknowledgements. The authors would like to thank Bakhadyr Khoussainov for helpful conversations about the material in this paper. The second author would also like to thank Hilary Craig for all her help and encouragement.

References

[1] W. Baur, G. Cherlin and A. Macintyre, Totally categorical groups and rings, *J. Algebra* **57** (1979), 407–440.

[2] A. Blumensath, *Automatic Structures* (Diploma Thesis, University of Aachen, 1999).

[3] A. Blumensath and E. Grädel, Finite presentations of infinite structures: automata and interpretations, *in* G. Gottlob, M. Hermann & M. Rusinowitch (eds.), *Proceedings of the 2nd International Workshop on Complexity in Automated Deduction* (2002).

[4] D. Cenzer and J. Remmel, Polynomial time versus recursive structures, *Annals Pure Appl. Logic* **54** (1991), 17–58.

[5] C. Delhommé, V. Goranko and T. Knapik, Automatic linear orderings (preprint, 2003).

[6] D. B. A. Epstein, J. W. Cannon, D. F. Holt, S. V. F. Levy, M. S. Paterson and W. P. Thurston, *Word Processing in Groups* (Jones and Bartlett, 1992).

[7] Ju. L. Eršov, Elementary group theories (Russian), *Dokl. Akad. Nauk SSR* **203** (1972), 1240–1243; English translation *Soviet Math. Dokl.* **13** (1972), 528–532.

[8] R. I. Grigorchuk, On growth in group theory, *Proceedings of the International Congress of Mathematicians, Kyoto 1990, Volume I, II* (Math. Soc. Japan, 1991), 325–338.

[9] M. Gromov, Groups of polynomial growth and expanding maps, *Publications Mathématique d'IHÉS* **53** (1981), 53–78.

[10] W. Hodges, *Model Theory* (Encyclopedia of Mathematics and its Applications **42**, Cambridge University Press, 1993).

[11] B. Khoussainov and A. Nerode, Automatic presentations of structures *in* D. Leivant (ed.), *Logic and Computational Complexity* (Lecture Notes in Comp. Sci. **960**, Springer-Verlag, 1995), 367–392.

[12] B. Khoussainov, A. Nies, S. Rubin and F. Stephan, Automatic structures: richness and limitations, *Proceedings of the 19th IEEE Symposium on Logic in Computer Science* (IEEE Computer Society, 2004), 110–119.

[13] B. Khoussainov and S. Rubin, Some thoughts on automatic structures, *J. Autom. Lang. Comb.* **8** (2003), 287–301.

[14] B. Khoussainov, S. Rubin and F. Stephan, On automatic partial orders, *Proceedings of the 18th IEEE Symposium on Logic in Computer Science* (IEEE Computer Society, 2003), 168–177.

[15] G. A. Noskov, The elementary theory of a finitely generated almost solvable group (Russian), *Izv. Akad. Nauk SSSR Ser. Mat.* **47** (1983), 498–517; English translation *Math. USSR Izv.* **22** (1984), 465–482.

[16] N. S. Romanovskiĭ, On the elementary theory of an almost polycyclic group (Russian), *Math. Sb.* **111** (1980), 135–143; English translation *Math. USSR Sb.* **39** (1981).

[17] S. Rubin, Finite automata and well ordered sets, *in* S. Yeates (ed.), *Third New Zealand Computer Science Research Students' Conference, Hamilton, New Zealand* (University of Waikato, 1999), 86–93.