The Isomorphism Problem for Plain Groups Is in Σ_3^{P}

Heiko Dietrich
Monash University, Clayton, Australia

Murray Elder[1]
University of Technology Sydney, Ultimo, Australia

Adam Piggott
Australian National University, Canberra, Australia

Youming Qiao
University of Technology Sydney, Ultimo, Australia

Armin Weiß
Universität Stuttgart, Germany

Abstract

Testing isomorphism of infinite groups is a classical topic, but from the complexity theory viewpoint, few results are known. Sénizergues and the fifth author (ICALP2018) proved that the isomorphism problem for virtually free groups is decidable in PSPACE when the input is given in terms of so-called virtually free presentations. Here we consider the isomorphism problem for the class of *plain groups*, that is, groups that are isomorphic to a free product of finitely many finite groups and finitely many copies of the infinite cyclic group. Every plain group is naturally and efficiently presented via an inverse-closed finite convergent length-reducing rewriting system. We prove that the isomorphism problem for plain groups given in this form lies in the polynomial time hierarchy, more precisely, in Σ_3^{P}. This result is achieved by combining new geometric and algebraic characterisations of groups presented by inverse-closed finite convergent length-reducing rewriting systems developed in recent work of the second and third authors (2021) with classical finite group isomorphism results of Babai and Szemerédi (1984).

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Rewrite systems; Mathematics of computing → Discrete mathematics; Theory of computation → Computability

Keywords and phrases plain group, isomorphism problem, polynomial hierarchy, Σ_3^{P} complexity class, inverse-closed finite convergent length-reducing rewriting system

Digital Object Identifier 10.4230/LIPIcs.STACS.2022.26

Funding *Heiko Dietrich*: Supported by Australian Research Council grant DP190100317.
Murray Elder: Supported by Australian Research Council grant DP210100271.
Adam Piggott: Supported by Australian Research Council grant DP210100271.
Youming Qiao: Supported by Australian Research Council grant DP200100950.

Acknowledgements We wish to thank the reviewers for their helpful comments and corrections.

1 Introduction

The classical core of combinatorial group theory centres on Dehn's three algorithmic problems concerning finitely presented groups [7]: given a finite presentation for a group, describe an algorithm that decides whether or not an arbitrary word in the generators and their

[1] Corresponding author

inverses spells the identity element (the word problem); given a finite presentation for a group, describe an algorithm that decides whether or not two arbitrary words in the generators and their inverses spell conjugate elements (the conjugacy problem); describe an algorithm that, given two finite presentations, decides whether or not the groups presented are isomorphic (the isomorphism problem). For arbitrary finite presentations, these problems are undecidable and so upper bounds on complexity are impossible. Obtaining bounds on complexity requires working with presentations that come with a promise that they determine groups in a particular class, or presentations that provide (intrinsic or extrinsic) additional structure. From Dehn's work, finite length-reducing rewriting systems that satisfy various convergence properties emerged as finite group presentations that simultaneously specify interesting infinite groups and provide natural solutions to the corresponding word problems.

A research program, active since the 1980s, seeks to classify the groups that may be presented by finite length-reducing rewriting systems satisfying various convergence properties. The hyperbolic groups [15], the virtually-free groups [14] (groups with a free subgroup of finite index – by Muller and Schupp's theorem [23] they are also known as context-free groups), and the plain groups [10] are important classes of groups, each a proper subclass of the class before, that arise within this program. A group is *plain* if it is isomorphic to a free product of finitely many finite groups and a free group of finite rank. The plain groups may be characterised as the fundamental groups of finite graphs of finite groups with trivial edge groups [18], and as the groups admitting a finite group presentation with a simple reduced word problem [16]. Moreover, the plain groups are conjectured to be exactly the groups that may be presented by finite convergent length-reducing rewriting systems [21].

The isomorphism problem is, of course, the most difficult of Dehn's problems and complexity results concerning this problem are rare. However, progress has been made on the isomorphism problem for the very classes of groups that arise in the study of length-reducing rewriting systems. Krstić solved the isomorphism problem for virtually-free groups described by arbitrary finite group presentations [19]. Building on the pioneering work of Rips and Sela [26], Sela [27], and Dahmani and Groves [5], Dahmani and Guirardel [6] provided an explicit algorithm that solves the isomorphism problem in all hyperbolic groups when the groups are given by finite presentations. In light of this result, attention can now shift to complexity bounds for the isomorphism problem. Notice that, in order to obtain complexity bounds, we cannot allow arbitrary presentations as inputs (otherwise we could decide within that complexity bound whether a given presentation is for the trivial group – a problem which is undecidable). In [28, 29], Sénizergues showed that the isomorphism problem for virtually-free groups is primitive recursive when the input is given in the form of two virtually-free presentations, or as two context free grammars. A virtually-free presentation of a group G specifies a free subgroup F plus a set of representatives S for the cosets $F \setminus G$ together with relations describing pairwise multiplications of elements from F and S; a context-free grammar can specify a virtually-free group by generating the language of words that spell the identity element. Then in 2018, Sénizergues and the fifth author [30] showed that the isomorphism problem for virtually-free groups can be solved in doubly-exponential space when the groups are specified by context-free grammars, and in PSPACE when the groups are given by virtually-free presentations.

In the present article, we prove that the complexity bounds for the isomorphism problem in virtually-free groups can be improved significantly when one restricts attention to the class of plain groups.

▶ **Theorem 1** (Isomorphism of plain groups). *The isomorphism problem for plain groups presented by inverse-closed finite convergent length-reducing rewriting systems is in* Σ_3^{P}.

We recall that the complexity class Σ_3^{P} lies in the *polynomial hierarchy*, see for example [1, Chapter 5]:

$$\mathsf{NP} = \Sigma_1^{\mathsf{P}} \subseteq \Sigma_2^{\mathsf{P}} \subseteq \Sigma_3^{\mathsf{P}} \cdots \subseteq \mathsf{PSPACE}.$$

Here we use the following specific definition.

▶ **Definition 2** ([32]). *Let $\mathcal{S}$ be a finite set and $L \subseteq \mathcal{S}^*$. Then $L \in \Sigma_3^{\mathsf{P}}$ if and only if there is a polynomial p and a predicate P that can be evaluated in* PTIME *such that*

$$\forall w \in \mathcal{S}^* \left(w \in L \iff \exists x \in \{0,1\}^{p(|w|)} \, \forall y \in \{0,1\}^{p(|w|)} \, \exists z \in \{0,1\}^{p(|w|)} P(w,x,y,z) = 1 \right).$$

Rather than specifying the variables as polynomial length binary strings, we will describe data for x, y, z which has polynomial size over a finite alphabet.

▶ Remark 3. We will abbreviate *inverse-closed finite convergent length-reducing rewriting system* to *icfclrrs* for the rest of this article, and refer to a group admitting a presentation by an icfclrrs as an *icfclrrs group*.

▶ Remark 4. In [11] it is shown that the problem of deciding if an icfclrrs presents a plain group is in NP. Note that if the conjecture that inverse-closed finite convergent rewriting systems can only present plain groups is proved, the word "plain" may be omitted from the statement of Theorem 1.

Note that given an icfclrrs for a group, one can compute a context-free grammar for the word problem in polynomial time using the method from [8]. Hence, the results from [30] imply a doubly-exponential-space algorithm for our situation. Therefore, Theorem 1 represents a significant improvement for this special case. In [11] the second and third authors gave a bound of PSPACE for isomorphism of plain groups given as icfclrrss. This PSPACE algorithm builds upon new geometric and algebraic characterisations of icfclrrs groups developed in the same paper. Theorem 1 is again a significant improvement on this, lowering the complexity to the third level of the polynomial hierarchy. The proof of Theorem 1 combines the new characterisations of icfclrrs groups from [11], which enable us to understand maximal finite subgroup structure and conjugacy of finite order elements in these groups, with work of Babai and Szemerédi [3] to test isomorphism of finite groups efficiently using straight-line programs.

Let us briefly give a high-level intuition of the proof of Theorem 1. Verifying the ranks of the free factors of each group are the same is straightforward, so for this brief description let us assume the two plain groups are simply free products of n finite groups. We existentially guess generating sets $\mathscr{A}_1, \ldots, \mathscr{A}_n$ and $\mathscr{B}_1, \ldots, \mathscr{B}_n$ for the finite factors in each group such that $|\mathscr{A}_i| = |\mathscr{B}_i|$ and mapping each $\mathscr{A}_i$ to $\mathscr{B}_i$ defines an isomorphism (in other words, we guess an isomorphism defined on generating sets), then, using straight-line programs (and methods from [3]), we universally verify that our guess indeed defines an isomorphism (that $\phi(g)\phi(h) = \phi(gh)$ for all g, h). Technically this is an infinite universal branching – still, the results of [11] ensure that considering polynomial-length straight-line programs suffice to verify that we guessed an isomorphism correctly.

However, be aware that we also need to verify that the sets we guessed, indeed, generate each group – and this is actually the more difficult part. In order to do so, we check that every element of finite order (universal branching) is conjugate to an element in the subgroup generated by some $\mathscr{A}_i$ (existential branching). Again by results in [11] we can restrict to polynomial-length straight-line programs in both the universal and existential branching. Moreover, testing whether an element has finite order can be done in polynomial time. This leads to a Σ_3^{P} algorithm.

Outline. The article is organised as follows. In Section 2, we provide background information on rewriting systems, and state the key algebraic results from [11] we need for the present work. In Section 3, we review the necessary background on straight-line programs for groups. In Section 4, we formulate a result which allows us to verify when two finite subgroups of two (potentially infinite) groups are related by an isomorphism, based on a result of Babai and Szemerédi [3]. Section 5 is devoted to a proof of our main result, Theorem 1.

Notation. Throughout this article we write log to mean $\log_2$. For $n \in \mathbb{N}_+$ we write $[1, n]$ for the interval $\{1, \ldots, n\} \subseteq \mathbb{N}$. If $\mathcal{S}$ is an *alphabet* (a non-empty finite set), we write $\mathcal{S}^*$ for the set of finite-length words over $\mathcal{S}$, and $|u|$ for the length of the word $u \in \mathcal{S}^*$; the empty word, λ, is the unique word of length 0. For a group G, we write e_G for the identity element of G.

2 Finite convergent length-reducing rewriting systems

2.1 Rewriting systems and groups

Let $\mathcal{S}$ be a generating set for a group G. If $v, w \in (\mathcal{S} \cup \mathcal{S}^{-1})^*$ and $g, h \in G$, then we write $v =_G g$ if the product of letters in v equals g; we write $v = w$ if v and w are identical as words, and $g = h$ if g and h represent the same element of G. If $v =_G g$ we say that v *spells* g. For example, the identity element e_G is spelled by the empty word λ, by aa^{-1} for any $a \in \mathcal{S}$, and so on. For an integer $r \geqslant 0$, we define the *ball of radius r in G with respect to the generating set $\mathcal{S}$*, denoted as $B_{e_G}(r)$, to be the set of all elements $g \in G$ for which there exists a word in $(\mathcal{S} \cup \mathcal{S}^{-1})^*$ of length at most r that spells g. For example, if G is the free abelian group $\langle a, b \mid ab = ba \rangle$ generated by $\mathcal{S} = \{a, b, a^{-1}, b^{-1}\}$ then the ball of radius 2 is the set of thirteen elements

$$\{e_G, a, b, a^{-1}, b^{-1}, a^2, ab, b^2, a^{-1}b, a^{-2}, a^{-1}b^{-1}, b^{-2}, ab^{-1}\}.$$

We briefly recall some basic facts concerning finite convergent length-reducing rewriting systems necessary for our discussion. We refer the reader to [4] for a broader introduction. A length-reducing rewriting system is a pair $(\mathcal{S}, T)$, where $\mathcal{S}$ is a non-empty alphabet, and T is a subset of $\mathcal{S}^* \times \mathcal{S}^*$, called a set of *rewriting rules*, such that for all $(\ell, r) \in T$ we have that $|\ell| > |r|$. We write $\not\sim_T = \max_{(\ell,r)\in T}\{|r|\}$.

The set of rewriting rules determines a relation $\to$ on the set $\mathcal{S}^*$ as follows: $a \to b$ if $a = u\ell v$, $b = urv$, and $(\ell, r) \in T$. The reflexive and transitive closure of $\to$ is denoted $\xrightarrow{*}$. A word $u \in \mathcal{S}^*$ is *irreducible* if no factor is the left-hand side of any rewriting rule, and hence $u \xrightarrow{*} v$ implies that $u = v$.

The reflexive, transitive and symmetric closure of $\to$ is an equivalence denoted $\overset{*}{\leftrightarrow}$. The operation of concatenation of representatives is well defined on the set of $\overset{*}{\leftrightarrow}$-classes, and hence makes a monoid $M = M(\mathcal{S}, T)$. We say that M is the *monoid presented by* $(\mathcal{S}, T)$. When the equivalence class of every letter (and hence also the equivalence class of every word) has an inverse, the monoid M is a group and we say it is *the group presented by* $(\mathcal{S}, T)$. We note that if a rewriting system $(\mathcal{S}, T)$ presents a group G, then $\langle \mathcal{S} \mid \ell = r \text{ for } (\ell, r) \in T \rangle$ is a group presentation for G. We say that $(\mathcal{S}, T)$ (or just $\mathcal{S}$) is *inverse-closed* if for every $a \in \mathcal{S}$, there exists $b \in \mathcal{S}$ such that $ab \xrightarrow{*} \lambda$. Clearly, M is a group when $\mathcal{S}$ is inverse-closed.

A rewriting system $(\mathcal{S}, T)$ is *finite* if $\mathcal{S}$ and T are finite sets, and *terminating* (or *noetherian)* if there are no infinite sequences of allowable factor replacements. It is clear that length-reducing rewriting systems are terminating. A rewriting system is *confluent* if whenever $w \xrightarrow{*} x$ and $w \xrightarrow{*} y$, there exists $z \in \mathcal{S}^*$ such that x and y both reduce to z. A

rewriting system is called *convergent* if it is terminating and confluent. In some literature, finite convergent length-reducing rewriting systems are called finite *Church-Rosser Thue* systems.

We define the *size* of a rewriting system $(\mathcal{S}, T)$ to be $n_T = |\mathcal{S}| + \sum_{(\ell,r)\in T} |\ell r|$, and we note that $r_T \leqslant n_T$.

2.2 Plain groups represented as rewriting systems

If $G_1, \ldots, G_n$ are groups with each G_i presented by $\langle \mathcal{S}_i \mid R_i \rangle$ for pairwise disjoint $\mathcal{S}_1, \ldots, \mathcal{S}_n$, the *free product* $G_1 * \cdots * G_n$ is the group presented by $\langle \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_n \mid R_1 \cup \cdots \cup R_n \rangle$. A group is *plain* if it is isomorphic to the free product

$$A_1 * A_2 * \cdots * A_p * F_r$$

where p, r are non-negative integers, each A_i is a finite group and F_r is the free group of rank r.

We first observe that every plain group admits a presentation by an icfclrrs (see for example [12, Corollary 2]).

▶ **Lemma 5.** *If G is a plain group, then G admits a presentation by a finite convergent length-reducing rewriting system $(\mathcal{S}, T)$ such that $\mathcal{S} = \mathcal{S}^{-1}$ and the left-hand side of every rule has length 2.*

The following fact follows easily from the normal form theory of free products (see for example [20]).

▶ **Lemma 6.** *Two plain groups given as*

$$A_1 * A_2 * \cdots * A_p * F_r \quad \text{and} \quad B_1 * B_2 * \cdots * B_q * F_s$$

are isomorphic if and only if $p = q, r = s$ and there is a permutation σ such that $A_i \cong B_{\sigma(i)}$ for every $i \in [1, p]$.

The following proposition collects key results about icfclrrs groups proved in [11]. Recall the definitions of r_T, n_T above.

▶ **Proposition 7** ([11, Proposition 15, Lemmas 12, 8, 18])**.** *If G is a plain group presented by an icfclrrs $(\mathcal{S}, T)$, then*

1. *every finite subgroup H of G is conjugate to a subgroup in $B_{e_G}(r_T + 2)$;*
2. *the number of conjugacy classes of maximal finite subgroups in G is bounded above by n_T^2;*
3. *if $g, h \in B_{e_G}(r_T + 2) \setminus \{e_G\}$ are conjugate elements of finite order and $t \in G$ is such that $tgt^{-1} = h$, then $t \in Be_G(5r_T + 4)$;*
4. *$\log(|B_{e_G}(r_T + 2)|) \leqslant n_T^2$.*

We also make use of the following facts about finite subgroup membership.

▶ **Lemma 8** ([11, Lemma 11])**.** *Let G be a plain group. For $g, h \in G$ define $g \sim h$ if gh has finite order. Then*

1. *the relation $\sim$ is transitive on the set of non-trivial finite-order elements in G;*
2. *a set $\mathcal{A} = \{a_1, \ldots, a_m\} \subseteq G \setminus \{e_G\}$ generates a finite subgroup if and only if for all $i \in [1, m]$ both a_i and $a_1 a_i$ have finite order;*
3. *if A is a finite subgroup of G and $g, h \in G$ with $g \in A \setminus \{e_G\}$, then $h \in A$ if and only if h and gh have finite order.*

2.3 Algorithms for groups in rewriting systems

Next we observe that deciding if elements have finite order can be done in polynomial time.

▶ **Lemma 9** (Narendran and Otto [24, Theorem 4.8]). *There is a deterministic polynomial-time algorithm for the following problem: given an icfclrrs $(\mathcal{S}, T)$ presenting a group G and a word $u \in \mathcal{S}^*$, decide whether or not u spells an element of finite order in G. The running time is polynomial in $|T| + |u| + \sum_{(r,\ell)\in T} |\ell|$, so polynomial in $|u| + n_T$.*

By computing the Smith normal form of a matrix associated to $(\mathcal{S}, T)$, we have an efficient way to compute the number of infinite cyclic factors of a plain group given by an icfclrrs.

▶ **Lemma 10.** *There is a deterministic polynomial-time algorithm for the following problem: given an icfclrrs $(\mathcal{S}, T)$ presenting a group G, compute the torsion-free rank of the abelian group $G/[G, G]$. The running time is polynomial in n_T, and the torsion-free rank is bounded above by n_T.*

Proof. Let G_{ab} denote $G/[G, G]$, the abelianization of G. Let r denote the *torsion-free rank* of G_{ab}, which is the number of factors $\mathbb{Z}$ in the free product decomposition of G. We may compute the torsion-free rank of the abelianization G_{ab} from $(\mathcal{S}, T)$ in time that is polynomial in n_T as follows. Let $\mathcal{S}' \subseteq \mathcal{S}$ be a subset comprising exactly one generator from each pair of inverses. The information in $(\mathcal{S}, T)$ may be recorded in the form of a group presentation $\langle \mathcal{S}' \mid R \rangle$, where R interprets each rewriting rule in T as a relation over the alphabet $\mathcal{S}' \cup (\mathcal{S}')^{-1}$. The information in the presentation $\langle \mathcal{S}' \mid R \cup \{[a, b] \mid a, b \in \mathcal{S}'\}\rangle$ for G_{ab} may be encoded in an $|R| \times |\mathcal{S}'|$ matrix of integers M. These integers record the exponent sums of generators in each relation. The Smith normal form matrix S corresponding to M may be computed in time that is polynomial in the size of the $|R| \times |\mathcal{S}'|$ matrix and its entries (see, for example, [17, 33]), so polynomial in n_T. The torsion-free rank of G_{ab} is the number of zero entries along the diagonal of S (see, for example, [25, pp. 376-377]). Note that this means $r \leqslant n_T$. ◀

3 Straight-line programs

We use *straight-line programs* (or more precisely *straight-line sequences*) to represent the elements of a group A with finite generating set $\mathscr{A} = \{a_1, \ldots, a_m\}$, see [31, Section 1.2.3] or [3, Section 3] for more details; we briefly recall this concept here. Let $X = \{x_1, \ldots, x_m\}$ be a set of abstract symbols of size m. A straight-line program Y of rank m and length d on X is a sequence $Y = (s_1, \ldots, s_d)$ where for each $i \in [1, d]$ either $s_i \in X \cup \{\lambda\}$, or $s_i = s_j s_k$ for some $j, k < i$, or $s_i = s_j^{-1}$ for some $j < i$. One says the straight-line program Y *yields* the word $w = s_d \in (X \cup X^{-1})^*$, which we also denote by $Y(x_1, \ldots, x_m) = w(x_1, \ldots, x_m)$. We write $Y(a_1, \ldots, a_m) \in (\mathscr{A} \cup \mathscr{A}^{-1})^*$ for the word that is constructed by replacing every occurrence of $x_i^{\pm 1}$ in $Y(x_1, \ldots, x_m)$ by $a_i^{\pm 1}$. We call the element $g \in A$ such that $Y(a_1, \ldots, a_m) =_G g$ the *evaluation* of Y in A (with respect to $\mathscr{A}$).

An efficient way to store the straight-line program is to write instead the operations that define the elements $s_1, \ldots, s_m$ of the sequence (cf. [31, p. 10]): for example, a generator $s_i = x$ is stored as the pair $(x, +)$, $s_i = \lambda$ is stored as $(\lambda, +)$, an inverse $s_i = s_j^{-1}$ is stored as $(j, -)$, and a product $s_i = s_j s_k$ is stored as (j, k). We call this sequence of operations a *straight-line sequence*. The word $Y(a_1, \ldots, a_m)$ can then be computed by following the construction described in this straight-line sequence and replacing every generator x_j by a_j. To store this sequence we simply store the address (an integer in $[1, d]$ in binary) and the instruction (an integer in $[1, m]$ or at most two integers in $[1, d]$ in binary); thus a straight-line

sequence of rank m and length d requires $\mathcal{O}(d(\log(d)+\log(m)))$ bits. In what follows, a straight-line program will always be represented by a straight-line sequence, and we write Y both for a straight-line program and the straight-line sequence representing it.

▶ **Example 11.** Consider the infinite cyclic group G generated by $\mathscr{A}=\{a\}$. The straight-line sequence $Y=(y_0=(x,+),y_1=(0,0),y_2=(1,1),y_3=(2,2),y_4=(3,3),y_5=(4,2),y_6=(5,0),y_7=(6,-))$ yields the word $Y(x)=x^{-21}$ in $(X\cup X^{-1})^*$ with $X=\{x\}$, and $Y(a)$ yields the element a^{-21} of G. The straight-line sequence $Y=((\lambda,+))$ yields $Y(x)=\lambda$, so $Y(a)=_G e_G$.

Every element of a finitely generated group with finite generating set $\mathscr{A}$ can be described by a straight-line sequence: one could first list $\mathscr{A}\cup\mathscr{A}^{-1}$ using $y_{2i-1}=(x_i,+)$ and $y_{2i}=(2i-1,-)$ for $i\in[1,|\mathscr{A}|]$, then choose a word that spells the desired element, and finally construct it letter-by-letter using $y_k=(k-1,j)$ (where $j=2i-1$ if the next letter is x_i, and $j=2i$ if the next letter is x_i^{-1}). However, Example 11 demonstrates that we can sometimes be more efficient than that. In fact, the following result shows that elements of a finite group always have short straight-line sequences, with respect to any given generating set.

▶ **Lemma 12** (Babai and Szemerédi [3, Lemma 7], Babai [2]). *Let A be a finite group with generating set $\mathscr{A}=\{a_1,\dots,a_m\}$. For each $g\in A$, there exists a straight-line sequence Y of rank m and of length at most $(\log|A|+1)^2$ such that $Y(a_1,\dots,a_m)=_G g$.*

If $P=(p_1,\dots,p_c),Q=(q_1,\dots,q_d)$ are two straight-line sequences of rank m and length c,d respectively, then we use the notation $[PQ]$ to denote the straight-line sequence of rank m and length $c+d+1$ defined as

$$[PQ]=(p_1,\dots,p_c,q_1\dots,q_d,(c,c+d)).$$

We call this the *product* of P and Q, since by construction if $P(x_1,\dots,x_m)=u$ and $Q(x_1,\dots,x_m)=v$ then $[PQ](x_1,\dots,x_m)=uv$. We denote by $[PQR]$ the straight-line sequence $[[PQ]R]$ of rank m and length $c+d+e+2$ where R has rank m and length e.

3.1 Compressed word problem

We note that in the setting of groups presented by icfclrrss, we can efficiently solve the word problem when the input is a straight-line sequence representing a group element.

▶ **Lemma 13** (Compressed word problem). *There is a deterministic algorithm for the following problem: given an icfclrrs $(\mathcal{S},T)$ presenting a group G, a set $\mathscr{A}=\{a_1,\dots,a_m\}\subseteq\mathcal{S}^*$ generating a subgroup $A\leqslant G$ such that $A\subseteq B_{e_G}(K)$ for some $K\in\mathbb{N}$, a word $u\in\mathcal{S}^*$ such that $u=_G g\in G$, and a straight-line sequence Y of rank m and length d, decide whether or not $Y(a_1,\dots,a_m)=_A g$.*

The running time is polynomial in $K+n_T+|u|+d+m+\max_i|a_i|$. In particular, if K is bounded by a polynomial in the input size, the algorithm runs in polynomial time.

Proof. For each $v\in\mathcal{S}^*$, let v^{-1} denote the formal inverse of v obtained by reversing and replacing each letter $x\in\mathcal{S}$ by $x^{-1}\in\mathcal{S}$.

Assume $Y=(y_1,\dots,y_d)$ where each $y_i=(x_j,+),(j,-)$ or (j,k). For $i\in[1,d]$ we compute and store a word $s_i\in\mathcal{S}^*$ of length at most K as follows:

- if $y_i=(x_j,+)$, set $s_i=a_j$;
- if $y_i=(j,-)$ with $j<i$, set $s_i=s_j^{-1}$;
- if $y_i=(j,k)$ for $j,k<i$, set s_i to be the reduced word obtained from s_js_k by applying rewriting rules.

Finally, return *true* if $s_d u^{-1}$ reduces to λ, and *false* otherwise.

Notice that that no s_i becomes longer than K. Therefore, each s_i can be computed in time polynomial in K plus the size of the rewriting system and the other data. ◀

4 Isomorphism testing for finite subgroups

In this section we describe an argument based on Babai and Szemerédi's work [3] which we require for proving Theorem 1. For now our setting is that we are given two groups (later these will be presented by rewriting systems) which come with efficient (polynomial time) algorithms to solve the word problem. Each group will contain some specified finite subgroup, say A in the first group and B in the second. We aim to verify in polynomial time the existence of an isomorphism from A to B. We start with the following well-known facts.

▶ **Lemma 14.** *Every finite group A has a generating set of size at most* $\log |A|$.

Proof. If $\{g_1, \ldots, g_m\}$ is a minimal generating set and A_n is the group generated by $\{g_1, \ldots, g_n\}$ for $n \in [1, m]$, then $|A_1| \geqslant 2$ and $|A_{n+1}| \geqslant 2|A_n|$ for $n \in [1, m-1]$, so $|A| \geqslant 2^m$ by induction. ◀

▶ **Lemma 15.** *Let A and B be groups. A map $f\colon A \to B$ is a group homomorphism if and only if $f(e_A) = e_B$ and $f(g)f(h)f((gh)^{-1}) = e_B$ for all $g, h \in A$*

Proof. If f is a homomorphism, then these conditions hold. Conversely, the second condition with $g = e_A$ yields $f(h)f(h^{-1}) = e_B$, so $f(h^{-1}) = f(h)^{-1}$ and $e_B = f(g)f(h)f((gh)^{-1}) = f(g)f(h)f(gh)^{-1}$. Thus, $f(g)f(h) = f(gh)$ for all $g, h \in G$. ◀

We now state the key technical result, which is the essence of [3, Proposition 4.8] where the isomorphism problem for finite groups in the so-called *black-box model* is shown to be in Σ_3^{P}.

▶ **Proposition 16** (Isomorphism between finite subgroups)**.** *Let A, B be finite groups and $K \in \mathbb{N}_+$. Let $\mathscr{A} = \{a_1, \ldots, a_m\}, \mathscr{B} = \{b_1, \ldots, b_m\}$ be generating sets for A, B respectively, with $m \leqslant K$.*

Assume that for each $g \in A$ there is a straight-line program Y_g of rank m and length at most K such that $Y_g(a_1, \ldots, a_m) =_A g$, and likewise for $g \in B$ there is a straight-line program Z_g of rank m and length at most K such that $Z_g(b_1, \ldots, b_m) =_B g$.

Then the map $\psi : \mathscr{A} \to \mathscr{B}$ with $a_i \mapsto b_i$ induces an isomorphism $A \to B$ if and only if

$$Y(a_1, \ldots, a_m) =_A e_A \iff Y(b_1, \ldots, b_m) =_B e_B \tag{1}$$

for every straight-line program Y of rank m and length at most $3K + 2$.

Proof. If ψ induces an isomorphism, clearly (1) holds for all straight-line programs.

For the converse, assume that (1) holds on all rank-m straight-line programs up to length $3K + 2$.

Without loss of generality, assume $Y_{e_A} = Z_{e_B} = ((\lambda, +))$ and

$$Y_{a_i}(x_1, \ldots, x_m) = Z_{b_i}(x_1, \ldots, x_m) = ((x_i, +))$$

for each $i \in [1, m]$. So $Y_{e_A}(a_1, \ldots, a_m) =_A e_A$, $Z_{e_B}(b_1, \ldots, b_m) =_B e_B$, $Y_{a_i}(a_1, \ldots, a_m) =_A a_i$ and $Z_{b_i}(b_1, \ldots, b_m) =_B b_i$ for $i \in [1, m]$.

Define a map $\phi\colon A \to B$ as follows: for $g \in A$, evaluate $Y_g(b_1, \ldots, b_m)$ to get an element $h \in B$, then set $\phi(g) = h$. Thus, ϕ maps each a_i to b_i.

First, by way of contradiction suppose ϕ is not a homomorphism. Since $\phi(e_A) = e_B$ by definition, Lemma 15 shows that there must exist $g, h \in A$ such that $\phi(g)\phi(h)\phi((gh)^{-1}) \neq e_B$. This means that in A we have

$$Y_g(a_1, \ldots, a_m)Y_h(a_1, \ldots, a_m)Y_{(gh)^{-1}}(a_1, \ldots, a_m) =_A gh(gh)^{-1} =_A e_A,$$

whereas in B we have

$$Y_g(b_1, \ldots, b_n)Y_h(b_1, \ldots, b_n)Y_{(gh)^{-1}}(b_1, \ldots, b_n) =_B \phi(g)\phi(h)\phi((gh)^{-1}) \neq_B e_B.$$

Let $Y = [Y_g Y_h Y_{(gh)^{-1}}]$ be the straight-line program of rank m and length at most $3K + 2$. Then Y contradicts our assumption that (1) holds on all rank-m straight-line programs up to length $3K + 2$. Thus ϕ is a homomorphism.

Next we show that ϕ is injective. If $g \in \ker \phi$, then $Y_g(b_1, \ldots, b_m)$ evaluates to e_B; by assumption, (1) holds on input Y_g, so $g =_A Y_g(a_1, \ldots, a_m) =_A e_A$ and ϕ is injective. So we have shown that ϕ is a monomorphism which satisfies $\phi : a_i \mapsto b_i$.

Repeating the preceding argument for $\phi' \colon B \to A$ defined as: for $g \in B$, evaluate $Z_g(a_1, \ldots, a_m)$ to get an element $h \in A$, then set $\phi'(g) = h$; we obtain a monomorphism ϕ' with $\phi' : b_i \mapsto a_i$. Since A, B are finite this implies that $|A| = |B|$ hence the monomorphism ϕ is an isomorphism, and since $\phi(a_i) = b_i$ for $i \in [1, m]$ we have that ϕ is the (unique) isomorphism induced by ψ. ◀

▶ Remark 17. Using Lemma 13, we can check condition (1) in Proposition 16 in polynomial time in groups presented by icfclrrss.

5 Proof of the main theorem

The algorithm for the proof of our main theorem checks the conditions of the following proposition. We remark that verifying that some collection of finite subgroups are maximal and that every finite order element is conjugate to an element in one of these maximal finite subgroups turns out to be the main bottleneck for the complexity of our algorithm. These are items (2) and (3) in the following proposition.

▶ **Proposition 18.** *Let G, H be plain groups presented by icfclrrs $(\mathcal{S}, T)$, $(\mathcal{S}', T')$ respectively. Then $G \cong H$ if and only if there are subgroups $A_i \leqslant G, B_i \leqslant H$ for $i \in [1, p]$ such that the following conditions (1)–(5) are satisfied:*

(1) *for each $i \in [1, p]$ we have $A_i \subseteq B_{e_G}(\rho_T + 2)$ and $B_i \subseteq B_{e_H}(\rho'_T + 2)$ (in particular, they are finite subgroups).*

(2) *each A_i (resp. B_i) is a maximal finite subgroup of G (resp. H).*

(3) *every $g \in G \setminus \{e_G\}$ (resp. $h \in H \setminus \{e_H\}$) of finite order can be conjugated into exactly one A_i (resp B_i).*

(4) *for each $i \in [1, p]$ we have $A_i \cong B_i$.*

(5) *the torsion-free rank of $G/[G, G]$ is equal to the torsion-free rank of $H/[H, H]$.*

Moreover, we may choose minimal generating sets $\mathscr{A}_i \subseteq B_{e_G}(\rho_T + 2)$, $\mathscr{B}_i \subseteq B_{e_H}(\rho'_T + 2)$ for A_i, B_i respectively, $i \in [1, p]$ so that for all $i \in [1, p]$:

(6) *$|\mathscr{A}_i| = |\mathscr{B}_i| = m_i \leqslant \log |A_i|$.*

(7) *if $\mathscr{A}_i = \{a_{i,j} \mid j \in [1, m_i]\}$, $\mathscr{B}_i = \{b_{i,j} \mid j \in [1, m_i]\}$, the map $a_{i,j} \mapsto b_{i,j}$ for $j \in [1, m_i]$ induces an isomorphism $A_i \to B_i$.*

Finally, we may replace conditions (1)–(3) by:

(8) *for every $g \in B_{e_G}(\rho_T + 2) \setminus \{e_G\}$ (resp. $h \in B_{e_H}(\rho'_T + 2) \setminus \{e_H\}$) and every $i \in [1, p]$, if g (resp. h) and $ga_{i,1}$ (resp. $hb_{i,1}$) have finite order, then $g \in A_i$ (resp. $h \in B_i$).*

(9) *every $g \in B_{e_G}(\rho_T + 2) \setminus \{e_G\}$ (resp. $h \in B_{e_H}(\rho'_T + 2) \setminus \{e_H\}$) of finite order can be conjugated into exactly one A_i (resp B_i); moreover, g can be conjugated into that A_i (resp B_i) by a conjugating element of length at most $5\rho_T + 4$ (resp. $5\rho'_T + 4$).*

(10) *for every $g \in B_{e_G}(\rho_T + 2) \setminus \{e_G\}$ (resp. $h \in B_{e_H}(\rho'_T + 2) \setminus \{e_H\}$), if g (resp. h) and $ga_{i,1}$ (resp. $hb_{i,1}$) have finite order, then $ga_{i,j}^{\epsilon} \in B_{e_G}(\rho_T + 2)$ (resp. $hb_{i,j}^{\epsilon} \in B_{e_H}(\rho'_T + 2)$) for every $j \in [1, m_i]$ and $\epsilon \in \{\pm 1\}$.*

Proof. First, assume that $G \cong H$. We will show that conditions (1)–(7) are satisfied. By Lemma 6 there exists an isomorphism $\psi\colon G \to H$ and free product decompositions

$$G \cong A_1 * A_2 * \cdots * A_p * F_r \quad \text{and} \quad H \cong B_1 * B_2 * \cdots * B_p * F_r$$

such that $A_i \cong B_i = \psi(A_i)$ for each $i \in [1, p]$. Moreover, by Proposition 7 (item 1) we may assume A_i, B_i each lie within the balls of radius $\rho_T + 2, \rho'_T + 2$ in the Cayley graphs of $(G, \mathcal{S}), (H, \mathcal{S}')$ respectively. By Lemma 14 there exist minimal generating sets $\mathscr{A}_i$ and $\mathscr{B}_i$ for A_i, B_i for $i \in [1, p]$ with $|\mathscr{A}_i| = |\mathscr{B}_i| \leqslant \log |A_i|$ (we may assume without loss of generality that we choose minimal generating sets to be of the same size). Since $\psi(A_i) = B_i$ we may without loss of generality choose generators so that condition (7) holds. The normal form theory for free products ([20]) gives that: for any $i \neq j$, $A_i \cap A_j = \{e_G\}$ (resp. $B_i \cap B_j = \{e_H\}$); if $p = 0$ then G and H are free groups, and if $p \neq 0$, there are exactly p conjugacy classes of non-trivial maximal finite subgroups in G (resp. H) and they are represented by $A_1, \ldots, A_p$ (resp. $B_1, \ldots, B_p$). Condition (3) follows immediately. Condition (5) follows immediately from the fact that $G/[G, G] \cong H/[H, H]$.

Conversely, suppose there are subgroups $A_i \leqslant G, B_i \leqslant H$ for $i \in [1, p]$ such that conditions (1)–(5) are satisfied. Conditions (2) and (3) give that every maximal finite subgroup in G (resp. H) is conjugate to exactly one of the subgroups $A_1, \ldots, A_p$ (resp. $B_1, \ldots, B_p$). Since G (resp. H) is a plain group, it follows that $G \cong A_1 * \cdots * A_p * F_r$ (resp. $H = B_1 * \cdots * B_p * F_s$) for some free group of rank r (resp. s). Condition (4) gives that $A_1 \cong B_1, \ldots, A_p \cong B_p$. Condition (5) gives that $r = s$ and $F_r \cong F_s$. Thus we have that $G \cong H$.

Now let us show that conditions (1)–(3) may be replaced by conditions (8)–(10). First suppose that conditions (1)–(7) are satisfied. Lemma 8 (item 3) implies condition (8). Condition (3) and Proposition (7) (item 3) imply condition (9). Condition (1) and Lemma 8(item 3) imply condition (10).

Now suppose that conditions (4)–(10) are satisfied. To establish that condition (10) implies condition (1), suppose A_i contains an element p which lies outside $B_{e_G}(\rho_T + 2)$. Let $u \in \mathscr{A}^*$ be a word spelling p. Then there exists a word u_1, an element $a_{i,j} \in \mathscr{A}$ and $\epsilon \in \{\pm 1\}$ so that $u_1 a_{i,j}^{\epsilon}$ is a prefix of u such that u_1 spells an element that lies in $B_{e_G}(\rho_T + 2)$ and $u_1 a_{i,j}^{\epsilon}$ spells an element that lies outside $B_{e_G}(\rho_T + 2)$. It follows that $u_1 \neq_G e_G$ (since $|a_{i,j}| \leqslant \rho_T + 2$). This contradicts condition (10). Conditions (1) and (8) together imply condition (2). Condition (9) and Proposition 7 (item 1) together imply condition (3). ◀

We are now ready to prove the main result.

Proof of Theorem 1. We describe a Σ_3^{P} algorithm which on input a pair $(\mathcal{S}, T)$, $(\mathcal{S}', T')$ of icfclrrss which are promised to present plain groups, accepts if and only if the groups are isomorphic. Let $N = \max\{n_T, n'_T\}$ be the input size, G the plain group presented by $(\mathcal{S}, T)$ and H the plain group presented by $(\mathcal{S}', T')$.

The algorithm needs to demonstrate the existence of some $p \in \mathbb{N}$ and subgroups $A_i \leqslant G$ and $B_i \leqslant H$ for $i \in [1,p]$ which satisfy conditions (4)–(10) of Proposition 18. We first observe the following. By Proposition 7 (item 2) there are at most n_T^2 (resp. $(n'_T)^2$) conjugacy classes of maximal finite subgroups in G (resp. H), so we have $p \leqslant N^2$. By Lemma 14 and Proposition 7 (item 4), if $\mathscr{A}$ (resp. $\mathscr{B}$) is a minimal generating set for a maximal finite subgroup A of G (resp. B of H), then

$$|\mathscr{A}| \leqslant \log|A| \leqslant \log(|B_{e_G}(r_T+2)|) \leqslant n_T^2 \leqslant N^2$$

(resp. $|\mathscr{B}| \leqslant (n'_T)^2 \leqslant N^2$). By Lemma 12, for each $g \in A$ (resp. $g \in B$), there exists a straight-line sequence Y of length at most $(\log|A|+1)^2 \leqslant N^4$ (resp. $(\log|B|+1)^2 \leqslant N^4$) such that Y yields g. Moreover, if $A \cong B$, we may assume they have minimal generating sets of the same size.

We now start with the following quantified statements:

$\exists$ sets $\mathscr{A}_i = \{a_{i,j} \in \mathcal{S}^* \mid j \in [1,m_i], |a_{i,j}| \leqslant r_T+2\}$,
$\mathscr{B}_i = \{b_{i,j} \in (\mathcal{S}')^*, \mid j \in [1,m_i], |b_{i,j}| \leqslant r'_T+2\}$
for $i \in [1,p]$ where $p \leqslant N^2, m_i \leqslant N^2$,

$\forall\ (u,v) \in \mathcal{S}^* \times (\mathcal{S}')^*$, $|u| \leqslant r_T+2, |v| \leqslant r'_T+2$,
$(s,s') \in \mathcal{S}^* \times (\mathcal{S}')^*$, $|s| \leqslant 5r_T+4, |s'| \leqslant 5r'_T+4$,
straight-line sequences Y_i of rank m_i and length at most $3N^4+2$ for each $i \in [1,p]$,

$\exists\ (t,t') \in \mathcal{S}^* \times (\mathcal{S}')^*$, $|t| \leqslant 5r_T+4, |t'| \leqslant 5r'_T+4$,
straight-line sequences Z_1, Z_2 of rank m_i for some $i \in [1,p]$ and length at most N^4.

Then the following procedure (predicate) verifies conditions (4)–(10) in Proposition 18 using this data.

First, apply Lemma 10 to compute the torsion-free rank of $G/[G,G]$ and $H/[H,H]$ and verify that the rank is the same for both. This establishes condition (5) of Proposition 18.

Next, run this subroutine:

for $i \in [1,p]$
  for $j \in [1,m_i]$
    verify that $a_{i,j}$ and $b_{i,j}$ have finite order using Lemma 9;
  for $j \in [2,m_i]$
    verify that $(a_{i,1}a_{i,j})$ and $(b_{i,1}b_{i,j})$ have finite order using Lemma 9.

This verifies that $\mathscr{A}_i, \mathscr{B}_i$ generate finite subgroups by Lemma 8 (item 2). Let A_i, B_i be the names of the subgroups generated by $\mathscr{A}_i, \mathscr{B}_i$ respectively. We can assume that the algorithm guesses the $\mathscr{A}_i, \mathscr{B}_i$ to be minimal generating sets of the same size, so we can assume that condition (6) is satisfied.

Next, we show that the finite subgroups A_i, B_i actually lie inside the ball of radius r_T+2 (resp. r'_T+2) by verifying condition (10) of Proposition 18. Run the following subroutine.

if u has finite order (using Lemma 9) and $u \neq_G e_G$ (reduced word for u is not λ)
  for $i \in [1,p]$
    if $(ua_{i,1})$ has finite order (using Lemma 9; if so then $u \in A_i$ by Lemma 8 (item 3))
      for $j \in [1,m_i]$
        compute the reduced word u_1 for $ua_{i,j}$ and u_2 for $ua_{i,j}^{-1}$,
        verify that $|u_1|, |u_2| \leqslant r_T+2$.

Repeat for the word v using the analogous procedure. This establishes condition (10).

Next, to verify condition (9) of Proposition 18, we first run this pre-step.

for $i \in [1,p]$
  for $k \in [1,p] \setminus \{i\}$
    verify that $(a_{i,1} s a_{k,1} s^{-1})$ and $(b_{i,1} s' b_{k,1} (s')^{-1})$ have infinite order using Lemma 9.

This shows that no conjugate of $a_{k,1}$ lies in A_i (resp. no conjugate of $b_{k,1}$ lies in B_i) for $i \neq k$ by Lemma 8 (item 3) (note that we are running over all s, s' of length at most $5\varkappa_T + 4, 5\varkappa'_T + 4$, so all elements in $B_{e_G}(5\varkappa_T + 4), B_{e_H}(5\varkappa'_T + 4)$).

Now suppose that for some $g \in G \setminus \{e_G\}$ we have $g =_G \alpha^{-1} c \alpha$ and $g =_G \beta d \beta^{-1}$ for some $\alpha, \beta, c, d \in \mathcal{S}^*$ with $c =_G g_c \in A_i, d =_G g_d \in A_k$ and $i \neq k$. Recall that as in Lemma 8, $g \sim h$ means gh has finite order. Then $c = (\alpha\beta) c (\alpha\beta)^{-1}$ so $a_{i,1} \sim (\alpha\beta) d (\alpha\beta)^{-1}$ and $(\alpha\beta) d (\alpha\beta)^{-1} \sim (\alpha\beta) a_{k,1} (\alpha\beta)^{-1}$, so by Lemma 8 (item 1) $a_{i,1} \sim (\alpha\beta) a_{k,1} (\alpha\beta)^{-1}$ which contradicts the result of the pre-step. It follows that every $g \in G \setminus \{e_G\}$ lies in a conjugate of *at most* one subgroup A_i. Thus to show condition (9) it suffices to show that every $u \in B_{e_G}(\varkappa_T + 2) \setminus \{e_G\}$ lies in a conjugate of *some* A_i (and analogously for v).

We show this with the following subroutine.

if u has finite order (using Lemma 9) and $u \neq_G e_G$ (reduced word for u is not λ)
  verify that $(u a_{i,1} t^{-1})$ has finite order for some $i \in [1,p]$
    (using Lemma 9 with a loop over all $i \in [1,p]$).

Repeat all of the above for the word v using the analogous procedure (using the word t'). This establishes condition (9) of Proposition 18.

Next, we verify condition (8) of Proposition 18. Run this subroutine.

if u has finite order (using Lemma 9) and $u \neq_G e_G$ (reduced word for u is not λ)
  for $i \in [1,p]$
    if $(u a_{i,1})$ has finite order (using Lemma 9)
      verify that $Z_1(a_{i,1} \ldots, a_{i,m_i}) =_G u$ using Lemma 13.

This shows that if $u \sim a_{i,1}$ then g can be spelled by a word in $(\mathscr{A}_i \cup \mathscr{A}_i^{-1})^*$, and so $u =_G g \in A_i$. Repeat for v using the analogous procedure (using the straight-line sequence Z_2). This establishes condition (8) of Proposition 18.

Lastly, to verify condition (7) and hence (4) of Proposition 18, we check that

$$Y_i(a_{i,1}, \ldots, a_{i,m_i}) = e_G \iff Y_i(b_{i,1}, \ldots, b_{i,m_i}) = e_H$$

holds where the Y_i are straight-line sequences of rank m_i and length at most $3N^4 + 2$ for $i \in [1,p]$ which we run through in the universal statement. This can be done in polynomial time using Lemma 13. Then by Proposition 16 (with $K = N^4$), since we are running over all straight-line sequences Y_i of length $3N^4 + 2$ and rank m_i for all $i \in [1,p]$ we establish condition (4). ◀

6 Conclusion

We have shown that the isomorphism problem for plain groups given as icfclrrss is decidable in Σ_3^{P}. To the best of our knowledge this presents the smallest complexity bound for the isomorphism problem apart from some very special cases like abelian and free groups (in polynomial time using [17, 25, 33]) and finite groups given as Cayley tables (in quasipolynomial time [13, 22] and also in NP, and nearly linear time for almost all orders [9]).

There is one obvious open question: can the complexity actually be reduced to Σ_2^{P} or even to some smaller class? Note that the obstacle to reach Σ_2^{P} is to verify condition (2) and (3) of Proposition 18 via conditions (8) and (9).

Another topic for future research is to investigate the maximal size of a finite subgroup presented by an icfclrrs. If one could show a polynomial bound on this size, rather than the exponential bound used here, this could lead to a lower complexity. However, this question is wide open – and, probably, related to the long-standing conjecture that all groups presented by icfclrrss are plain.

References

1 Sanjeev Arora and Boaz Barak. *Computational complexity*. Cambridge University Press, Cambridge, 2009. A modern approach. doi:10.1017/CBO9780511804090.

2 László Babai. Local expansion of vertex-transitive graphs and random generation in finite groups. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, STOC '91, pages 164–174, New York, NY, USA, 1991. Association for Computing Machinery. doi:10.1145/103418.103440.

3 László Babai and Endre Szemeredi. On the complexity of matrix group problems I. In *25th Annual Symposium on Foundations of Computer Science, 1984.*, pages 229–240, 1984. doi:10.1109/SFCS.1984.715919.

4 Ronald V. Book and Friedrich Otto. *String-rewriting systems*. Texts and Monographs in Computer Science. Springer-Verlag, New York, 1993. doi:10.1007/978-1-4613-9771-7.

5 François Dahmani and Daniel Groves. The isomorphism problem for toral relatively hyperbolic groups. *Publ. Math. Inst. Hautes Études Sci.*, 107:211–290, 2008. doi:10.1007/s10240-008-0014-3.

6 François Dahmani and Vincent Guirardel. The isomorphism problem for all hyperbolic groups. *Geom. Funct. Anal.*, 21(2):223–300, 2011. doi:10.1007/s00039-011-0120-0.

7 Max Dehn. *Papers on group theory and topology*. Springer-Verlag, New York, 1987. Translated from the German and with introductions and an appendix by John Stillwell, With an appendix by Otto Schreier. doi:10.1007/978-1-4612-4668-8.

8 Volker Diekert. Some remarks on presentations by finite Church-Rosser Thue systems. In *STACS 87 (Passau, 1987)*, volume 247 of *Lecture Notes in Comput. Sci.*, pages 272–285. Springer, Berlin, 1987. doi:10.1007/BFb0039612.

9 Heiko Dietrich and James B. Wilson. Group isomorphism is nearly-linear time for most orders, 2021. Accepted for FOCS 2021. arXiv:2011.03133.

10 Andy Eisenberg and Adam Piggott. Gilman's conjecture. *J. Algebra*, 517:167–185, 2019. doi:10.1016/j.jalgebra.2018.09.022.

11 Murray Elder and Adam Piggott. On groups presented by inverse-closed finite convergent length-reducing rewriting systems, 2021. arXiv:2106.03445.

12 Murray Elder and Adam Piggott. Rewriting systems, plain groups, and geodetic graphs. *Theoretical Computer Science*, 903:134–144, 2022. doi:10.1016/j.tcs.2021.12.022.

13 V. Felsch and J. Neubüser. On a programme for the determination of the automorphism group of a finite group. In Pergamon J. Leech, editor, *Computational Problems in Abstract Algebra (Proceedings of a Conference on Computational Problems in Algebra, Oxford, 1967)*, pages 59–60, Oxford, 1970.

14 Robert H. Gilman, Susan Hermiller, Derek F. Holt, and Sarah Rees. A characterisation of virtually free groups. *Arch. Math. (Basel)*, 89(4):289–295, 2007. doi:10.1007/s00013-007-2206-3.

15 Mikhail Gromov. Hyperbolic groups. In *Essays in group theory*, volume 8 of *Math. Sci. Res. Inst. Publ.*, pages 75–263. Springer, New York, 1987. doi:10.1007/978-1-4613-9586-7_3.

16 Robert H. Haring-Smith. Groups and simple languages. *Trans. Amer. Math. Soc.*, 279(1):337–356, 1983. doi:10.2307/1999388.

17 Ravindran Kannan and Achim Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM J. Comput.*, 8(4):499–507, 1979. doi:10.1137/0208040.

18 Abraham Karrass, Alfred Pietrowski, and Donald Solitar. Finite and infinite cyclic extensions of free groups. *Journal of the Australian Mathematical Society*, 16(4):458–466, 1973. doi:10.1017/S1446788700015445.

19 Sava Krstić. Actions of finite groups on graphs and related automorphisms of free groups. *J. Algebra*, 124(1):119–138, 1989. doi:10.1016/0021-8693(89)90154-3.

20 Roger C. Lyndon and Paul E. Schupp. *Combinatorial group theory*. Classics in Mathematics. Springer-Verlag, Berlin, 2001. Reprint of the 1977 edition. doi:10.1007/978-3-642-61896-3.

21 Klaus Madlener and Friedrich Otto. Groups presented by certain classes of finite length-reducing string-rewriting systems. In *Rewriting techniques and applications (Bordeaux, 1987)*, volume 256 of *Lecture Notes in Comput. Sci.*, pages 133–144. Springer, Berlin, 1987. doi:10.5555/30432.30444.

22 Gary L. Miller. On the $n^{\log n}$ isomorphism technique (a preliminary report). In *STOC*, pages 51–58, New York, NY, USA, 1978. ACM. doi:10.1145/800133.804331.

23 David E. Muller and Paul E. Schupp. The theory of ends, pushdown automata, and second-order logic. *Theoret. Comput. Sci.*, 37(1):51–75, 1985. doi:10.1016/0304-3975(85)90087-8.

24 Paliath Narendran and Friedrich Otto. Elements of finite order for finite weight-reducing and confluent Thue systems. *Acta Inform.*, 25(5):573–591, 1988.

25 Morris Newman. The Smith normal form. In *Proceedings of the Fifth Conference of the International Linear Algebra Society (Atlanta, GA, 1995)*, volume 254, pages 367–381, 1997. doi:10.1016/S0024-3795(96)00163-2.

26 Eliyahu Rips and Zlil Sela. Canonical representatives and equations in hyperbolic groups. *Invent. Math.*, 120(3):489–512, 1995. doi:10.1007/BF01241140.

27 Zlil Sela. The isomorphism problem for hyperbolic groups. I. *Ann. of Math. (2)*, 141(2):217–283, 1995. doi:10.2307/2118520.

28 Géraud Sénizergues. An effective version of Stallings' theorem in the case of context-free groups. In *Automata, languages and programming (Lund, 1993)*, volume 700 of *Lecture Notes in Comput. Sci.*, pages 478–495. Springer, Berlin, 1993. doi:10.1007/3-540-56939-1_96.

29 Géraud Sénizergues. On the finite subgroups of a context-free group. In *Geometric and computational perspectives on infinite groups (Minneapolis, MN and New Brunswick, NJ, 1994)*, volume 25 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 201–212. Amer. Math. Soc., Providence, RI, 1996. doi:10.1007/s002360050045.

30 Géraud Sénizergues and Armin Weiß. The isomorphism problem for finite extensions of free groups is in PSPACE. In *45th International Colloquium on Automata, Languages, and Programming*, volume 107 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 139, 14. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018.

31 Ákos Seress. *Permutation group algorithms*, volume 152 of *Cambridge Tracts in Mathematics*. Cambridge University Press, Cambridge, 2003. doi:10.1017/CBO9780511546549.

32 Larry J. Stockmeyer. The polynomial-time hierarchy. *Theoret. Comput. Sci.*, 3(1):1–22 (1977), 1976. doi:10.1016/0304-3975(76)90061-X.

33 Arne Storjohann. Near optimal algorithms for computing smith normal forms of integer matrices. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, ISSAC '96, pages 267–274, New York, NY, USA, 1996. Association for Computing Machinery. doi:10.1145/236869.237084.