Electronic Colloquium on Computational Complexity, Report No. 165 (2026)

# Group Isomorphism and the Polylogarithmic-Time Hierarchy: Depth-$2\frac{1}{2}$ Circuits and Lower Bounds

Joshua A. Grochow* Gülce Kardeş† Michael Levet‡

**Abstract**

In this paper, we investigate the low-depth circuit complexity of Group Isomorphism in the multiplication (Cayley) table model. We prove the first circuit lower bounds for Group Isomorphism: namely, we show that every family of depth-2 Boolean circuits deciding Group Isomorphism requires quasipolynomial-size. We complement this with upper bounds of uniform depth-$2\frac{1}{2}$ circuits of quasipolynomial-size.

A sequence of previous results from 1970–2025 progressively reduced the circuit depth from polynomial to $3\frac{1}{2}$; all of these results relied on the generator-enumerator strategy and, in fact, applied more generally to quasigroups. In contrast, our depth-$2\frac{1}{2}$ construction follows a fundamentally different strategy that exploits structure more specific to groups. We guess a composition series for each group, together with generators for its terms and the isomorphism types of its composition factors. We then inductively verify that the corresponding extensions at each level of the two composition series are compatible. A central part in this approach brings to bear the extensive work on the Short Presentation Conjecture, in tandem with the algorithmic theory of group extensions and cohomology.

## Contents



*Department of Computer Science, University of Colorado Boulder; Department of Mathematics, University of Colorado Boulder. Email: jgrochow@colorado.edu

†Department of Computer Science, University of Colorado Boulder; Santa Fe Institute. Email: gulce.kardes@colorado.edu

‡Department of Computer Science, College of Charleston. Email: levetm@cofc.edu

ISSN 1433-8092

# 1 Introduction

Isomorphism problems ask whether two objects are "the same" up to some natural notion of relabeling or transformation. They have played a central role in computational complexity, with Graph Isomorphism perhaps being the most well-studied (see [AD17] for more on its history and role in computational complexity). In this paper, we focus on its algebraic sibling, Group Isomorphism (GpI); in fact, since Group Isomorphism (when the input is given by multiplication tables) reduces to Graph Isomorphism [ZKT85], the former can be considered a particular algebraic sub-problem of the latter. From the algebraic perspective, the algorithmic Group Isomorphism problem is a very natural question of interest in its own right, having been asked by Dehn over a century ago, in 1912 [Deh12]; it has since become a central question in computational complexity and computational algebra.

We focus on the parallel, low-depth uniform circuit complexity of this problem when the inputs are given by their multiplication (Cayley) tables. In computer algebra systems, groups are almost always given succinctly either by a presentation, or generating sets of permutations or matrices, which are much more succinct than the Cayley table. However, even in these more succinct encodings, the runtime of the best-known algorithms for (finite) Group Isomorphism are $|G|^{\Theta(\log |G|)}$ [BEO02, ELGO02, BE99, CH03] (see [Wil19, p. 2]; see also [Ros13, Luk15] and [LGR16, Sec. 2.2] for the best-known worst-case analysis in the Cayley table model), a runtime that depends on the size of the group, but could be exponential relative to the more succinct input size. In contrast, in the Cayley table model $|G|$ is polynomially related to the input length, so the above running time is expressed (as a quasipolynomial) in terms of the input length.

In the Cayley table model, the $|G|^{O(\log |G|)}$ bound comes from the generator-enumerator technique [FN70, Mil78],[1] which has a short explanation: by Lagrange's Theorem, the order of a subgroup divides the order of the group, and a straightforward exercise then shows that every group $G$ is generated by at most $\log_2 |G|$ elements. The generator-enumerator isomorphism technique is to find a generating set of size $\leq \log |G|$ for $G$, then to try all generating sets of $H$ of the same size and to check whether one can map the generating set of $G$ to that of $H$ in a way that extends to an isomorphism. Since it was introduced in 1970, a series of results [FN70, Mil78, LSZ77, Wol94, Wag10, CTW13, Tan13, CGLW25] has shown that the generator-enumerator technique can be implemented in smaller and smaller complexity classes, albeit always with quasipolynomial time or circuit size (see Section 1.3 for details). The most recent advance [CGLW25] showed that this technique could be implemented by unbounded fan-in uniform circuits of $n^{O(\log n)}$ size and constant depth (the natural circuit class $\mathsf{quasiAC^0}$ [Bar92]), in fact depth only 4.[2]

Perhaps the biggest question about Group Isomorphism is whether it can be solved in polynomial time. From the parallel complexity perspective, the analogous big question is:

> Is Group Isomorphism in $\mathsf{AC^0}$?

We would be shocked if Group Isomorphism were in $\mathsf{AC^0}$. Despite this, circuit lower-bounds even at depth-2 have been elusive. In this paper, we make the first progress on circuit lower bounds for this question, as well as improve the depth of the upper bound (albeit still quasipolynomial size—that is, we do not improve the worst-case serial runtime). We cover these two results in each of the next two subsections.

[1] Miller attributes the result to Tarjan.

[2] Throughout the paper, unless otherwise noted, circuit classes denote their uniform versions.

### 1.1 Lower bounds

It was noted in [CGLW25] that we did not have lower bounds even against depth-2 circuits for GpI. And there is good reason for this difficulty: Chattopadhyay, Torán, and Wagner [CTW13] exhibited an upper bound of $\exists^{\log^2 n}\mathsf{FOLL} \cap \exists^{\log^2 n}\mathsf{L}$ for the more general Quasigroup Isomorphism problem (QGpI). Notably, $\exists^{\log^2 n}\mathsf{FOLL}$ cannot compute Parity or Majority [FSS84, CTW13]. Consequently, the standard strategy of obtaining $\mathsf{AC}^0$ lower bounds via reductions from Parity or Majority cannot work for GpI in this setting.

Our first main result is the first lower bound against Boolean circuits for GpI of which we are aware:

**Theorem A.** *Any (uniform or non-uniform) family of depth-2 circuits deciding* Group Isomorphism *requires size* $|G|^{\Omega(\log |G|)}$.

We show more strongly that DNFs require exponential size (Theorem 4.2), while CNFs require size $|G|^{\Omega(\log |G|)}$ (Theorem 4.3). Theorem A is thus the first step towards the longstanding open problem of whether GpI is in $\mathsf{AC}^0$ (see, e. g., [CTW13][3]).

This is in sharp contrast with the setting of Graph Isomorphism. where circuit lower bounds have long been known. To the best of our knowledge, the first complexity-theoretic lower bound for GI was established by Cai, Fürer, and Immerman [CFI92], who gave an $\mathsf{AC}^0$-reduction from Parity to GI. The best known lower bound for GI is $\mathsf{DET}$ [Tor04]. By contrast, although some lower bounds for GpI have been proved in semantically restricted models, we are not aware of prior lower bounds for Boolean circuits for GpI (see Related Work for more discussion).

**Methods.** Our proof of the CNF lower bound is significantly more involved than that for the DNF lower bound. The DNF lower bound follows from a relatively straightforward observation about Cayley tables of (quasi)groups: No two such Cayley tables can differ in exactly one entry. Our proof of the CNF lower bound combines (1) a novel reduction from Matrix Rank to GpI with (2) results by Meshulam [Mes85] and de Seguins Pazzis [dSP10] on the largest dimension of a(n affine) linear space of matrices of bounded rank. Because of the aforementioned result [CTW13], this reduction is necessarily not an $\mathsf{AC}^0$ reduction, which was an obstruction to previous attempts at such lower bounds. We get around this obstruction in that our reduction is an affine projection reduction: each output bit is an $\mathbb{F}_2$-affine linear combination of the input bits.

**Remark 1.1** (Lower bounds on Latin Square Isotopy)**.** Closely related to (Quasi)Group Isomorphism is the Latin Square Isotopy problem, which is essentially a special case of Strongly Regular Graph Isomorphism (see, e.g., [Mil78, Wol94]) that we now define. We say that two quasigroups are *isotopic* if there exist bijections $\alpha, \beta, \gamma : Q_1 \to Q_2$ such that for all $x, y, z \in Q_1$, $\gamma(xy) = \alpha(x)\beta(y)$. The Latin Square Isotopy problem takes as input two quasigroups $Q_1, Q_2$, and asks if they are isotopic. There has been some notable work on Latin Square Isotopy [Mil78, Wol94, Lev23, CGLW25]. In particular, Collins, Grochow, Levet, and Weiß [CGLW25] established the bound of $\exists^{\log^2 n}\forall^{\log n}\exists^{\log n}\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$, for Latin Square Isotopy, which is the same bound they established for Quasigroup Isomorphism. They also give a more careful analysis of their algorithm which implies that Latin Square Isotopy is decidable by a uniform family of depth-4 $\mathsf{quasiAC}^0$ circuits with size $n^{O(\log n)}$. Prior to our work, lower bounds

[3] As they state in their conclusion, their research "started originally trying to prove that QGroupIso is hard for $\mathsf{NC}^1$," yet to date we are not aware of any previous progress towards such a Boolean lower bound.

against polynomial-size, depth-2 circuits were open for Latin Square Isotopy, but they follow from our main results.

Namely, Albert [Alb43] established that two groups are isotopic if and only if they are isomorphic. Thus, in light of Theorem A, we obtain the following immediate corollary:

**Corollary 1.2.** *Any family of depth-2 circuits deciding* Latin Square Isotopy *requires size* $|G|^{\Omega(\log |G|)}$.

## 1.2 Depth-3 Circuits for Group Isomorphism

We complement our lower bound by also improving the depth of the upper bound on circuits for Group Isomorphism. To state our result, we recall that circuits of depth $d+1$ with bottom fan-in at most $\mathrm{poly}(\log n)$ are sometimes called "depth $d+\frac{1}{2}$" [BB05]; the aforementioned circuits [CGLW25] are in fact depth $3\frac{1}{2}$ in this nomenclature.

**Theorem B** (High-level version of Theorem 3.1)**.** Group Isomorphism *in the Cayley table model can be solved by uniform circuits of depth* $2\frac{1}{2}$ *and quasipolynomial size.*

*Furthermore, assuming the Uniform Short Presentation Conjecture, this can be improved to a conjunction of a quasipolynomial-size DNF and polynomial-size CNF.*

We actually obtain a slightly stronger upper bound, that places Group Isomorphism into the second level of the polylogarithmic-time hierarchy; see Theorem 3.1 for details. Collins, Grochow, Levet, and Weiß previously established that the more general Quasigroup Isomorphism problem belongs to the third level of the polylogarithmic time hierarchy, which yielded depth-$3\frac{1}{2}$ circuits. The trade-off compared to previous results is that they get size $n^{O(\log n)}$ [CGLW25], whereas our quasipolynomial size bound is slightly larger.

At a high level, rather than following the generator-enumerator strategy of previous results, our technique is instead to guess a polylogarithmic-length encoding of one group, and then verify that both groups satisfy the same encoding (hence must be isomorphic). As there are at most $n^{(2/27+o(1))\mu(n)^2}$ groups of order $\leq n$ [Pyb93, Hig60, Sim65], where $\mu(n) \leq \log_2 n$ is the maximum power of any prime-power divisor of $n$, it is at least information-theoretically plausible that groups of order $n$ could be specified by $O(\log^3 n)$ many bits.[4] As our strategy is to guess this specification—in terms of a circuit, that would be an OR gate of fan-in $2^{O(\log^3 n)}$—then in order to achieve a depth-3 circuit, we would also need to be able to verify those guesses, i. e., to check that the specification matches both groups, by a depth-3 circuit of quasipolynomial size that is an OR of ANDs of ORs, as then the top OR of the verification circuit can then be merged with the OR gate for the guesses.

(Already at this high level of description we can note a significant difference from the generator-enumerator approach: whereas the latter also works for Quasigroup Isomorphism, our approach here cannot give quasipolynomial size circuits for Quasigroup Isomorphism, by a standard counting argument; see Observation 1.4.)

We show that a *presentation* (in the group-theoretic sense) of a group can be verified as efficiently as needed for the above strategy to work. This uses the idea from [CGLW25] that basic group operations can be computed in deterministic polylogarithmic time from the multiplication table, and the fact that polylogarithmic time can be simulated by quasipolynomial-size depth-2 circuits. However, it is a long-open conjecture, known as the Short Presentation Conjecture, whether presentations of $O(\log^3 |G|)$ size—or even just $O(\log^c |G|)$ for any fixed $c$—exist [BS84, BGK+97].

[4] And in general this is asymptotically tight, as there are at least $p^{(2/27)m^3-O(m^2)}$ groups of order $p^m$ [Hig60, Sim65].

Babai, Goodman, Kantor, Luks, and Pálfy [BGK+97] introduced several strengthenings of the Short Presentation Conjecture that they anticipated would be useful for algorithmic applications. In particular, they conjectured not only did finite simple groups have such short presentations, but that those presentations could be constructed from the "standard names" (Def. 2.6) of the finite simple groups in polylogarithmic time (in the order of the group), which they refer to as the Uniform Short Presentation Conjecture [BGK+97, Conjecture 3].

**The status of the (Uniform) Short Presentation Conjecture.** Babai, Goodman, Kantor, Luks, and Pálfy proved [BGK+97, Theorem 8.3] that if finite simple groups admit presentations of size $O(\log^c n)$, then all groups admit presentations of size $O(\log^{c+1} n)$; exactly what we would hope for in order to get our strategy above to work to get depth-3 circuits. They also proved that the Uniform Short Presentation Conjecture held with $c = 2$ for all but three infinite families of finite simple groups [BGK+97],[5] namely the groups denoted

$$ {}^2A_2(q) = PSU_3(q), \qquad {}^2B_2(q) = Sz(q), \qquad \text{and} \qquad {}^2G_2(q) = R(q), $$

for infinitely many prime powers $q$.

The status of the Short Presentation Conjecture for finite simple groups is in a somewhat interesting state of affairs currently. Hulpke and Seress [HS01] showed that the first of the above three families satisfies the Short Presentation Conjecture with $c = 2$ (though they did not prove uniformity; we prove uniformity of their presentation in Section 5). They also remark that J. Thompson told W. Kantor that Suzuki's original paper [Suz62] on the groups ${}^2B_2(q) = Sz(q)$ (now known as the Suzuki groups) contained within it already a short presentation. Subsequently, Guralnick, Kantor, Kassabov, and Lubotzky [GKKL08] exhibited a shorter presentation using only $O(1)$ generators and relators for $Sz(q)$ (they actually established the analogous result for all finite simple groups except ${}^2G_2(q)$). We prove uniformity for this latter presentation in Section 6. This leaves only the Ree groups, ${}^2G_2(q) = R(q)$.

A short presentation for the Ree groups—and thus completion of the proof of the Short Presentation Conjecture—due to Hulpke, Kassabov, Seress (posthumously), and Wilson [HKSW20] has been announced in various talks and conferences since 2020, but has not yet appeared in print.

**More on our methods: avoiding the Short Presentation Conjecture for the Ree groups.** In order to get around this, we take advantage of two additional ingredients. The first is the $\Sigma_2\mathsf{TIME}(\text{polylog}(|G|))$ isomorphism test for black-box groups [BS84]. We use standard matrix generators to specify ${}^2G_2(q)$ (see, e. g., [Bää14]), and compare this to a composition factor $G_i/G_{i-1}$ from our non-deterministic guess. Precisely, this step allows us to decide whether $G_i/G_{i-1} \cong {}^2G_2(q)$, as well as whether the generators we guessed induce an isomorphism between $G_i/G_{i-1}$ and $H_i/H_{i-1}$ (the corresponding composition factor in $H$). If these checks pass, then it remains to decide whether the generators we guessed induce an isomorphism between $G_i \cong H_i$. The second ingredient is to solve the latter problem based on a computational version of non-Abelian cohomology [GQ17, Ded64, Ina97].

**Closer to depth 2 assuming the Uniform Short Presentation Conjecture.** If instead we *assume* the Uniform Short Presentation Conjecture, it not only avoids the latter ingredients, but

[5] This result, as all the similar results we mention, was based on a case analysis, and in that sense, when we say "all but three infinite families," the latter statement depends on the Classification of Finite Simple Groups.

we show it also implies that Group Isomorphism can be solved by a uniform family that is the conjunction of a quasipolynomial-size DNF with a polynomial-size CNF (see the second half of Theorem 3.1), even closer to depth 2. To see how much closer this is to depth 2, we offer two pieces of evidence: first, the resulting circuits are weft 2, whereas our unconditional depth-$2\frac{1}{2}$ circuits are weft 3—the weft is the maximum number of gates of unbounded fan-in encountered on any input-to-output path in the circuit, and is a key complexity measure in parametrized complexity [DF13]. Second, the circuits we get assuming the Uniform Short Presentation Conjecture are 2-tt (2 query, non-adaptive) projection reductions to quasi-polynomial size CNFs, whereas the circuits we get unconditionally have quasipolynomial top fan-in. If the 2-tt could be improved to a many-one projection reduction, the result would be a depth-2 circuit.

Following [CTW13], [CGLW25] highlighted the question of whether GPI is in $\exists^{\log^2 n}\mathsf{AC}^0$. A corollary of our main upper bound implies that the Uniform Short Presentation Conjecture answers this question positively, up to the exponent on the nondeterminism:

**Corollary 1.3** (=Corollary 3.2)**.** *The Uniform Short Presentation Conjecture implies* GPI $\in \exists^{\mathrm{polylog}(n)}\mathsf{AC}^0$.

**Difference compared to the generator-enumerator strategy.** As further technical evidence of the difference between our approach and the previous generator-enumerator results, we recall a significant obstacle to extending our approach from groups to quasigroups. Namely, a quasigroup analogue of the Short Presentation Conjecture is false, by a standard counting argument that we include here for completeness:

**Observation 1.4.** *Any injection from the set of isomorphism classes of quasigroups of order $n$ to binary strings of length $\ell$ must have $\ell \geq \Omega(n^2 \log n)$.*

In particular this says that quasigroups up to isomorphism cannot be specified, in the general case, more succinctly than by writing down their multiplication table, at least up to constant factors, regardless of the method of encoding (table, presentation, Turing machine, etc.).

*Proof.* The number of Latin squares of order $n$ is at least $(n!)^{2n}/n^{n^2}$ (e. g., [vLW01, Thm. 17.2]), hence the number of quasigroups of order $n$ up to isomorphism is at least

$$\begin{aligned}(n!)^{2n-1}/n^{n^2} &\sim \left((n/e)^n\sqrt{2\pi n}\right)^{2n-1}/n^{n^2}\\ &= n^{n^2-1/2}\sqrt{2\pi}^{2n-1}/e^{2n^2-n}\\ &\geq (n/c)^{\Theta(n^2)}\end{aligned}$$

for some constant $c \approx e^2/1.01$. Thus, in order for there to be at least as many strings of length $\ell$ as there are quasigroups of order $n$ (up to isomorphism), we must have $\ell \geq \log(n/c)^{\Theta(n^2)} \geq \Omega(n^2 \log n)$. □

## 1.3 Related Work

**Generator-enumeration strategy.** The generator-enumeration strategy was independently discovered by Felsch and Neubüser [FN70] and by Tarjan; see [Mil78]. Miller [Mil78] extended this observation to the setting of quasigroups, obtaining the $\exists^{\log^2 n}\mathsf{P}$ bound for Quasigroup Isomorphism, and much of the subsequent work on this strategy also works at that level of generality. We

summarize the history in Table 1, which is from [CGLW25] (except for adding a new entry for the present paper). We note that the upper bound of Chattopadhyay, Torán, Wagner [CTW13] was the first upper bound showing that PARITY did not $\mathsf{AC}^0$-reduce to GPI (allowing them to show that GPI did not $\mathsf{AC}^0$-reduce to GI).

Recently, Collins–Grochow–Levet–Weiß [CGLW25] exhibited a bound of

$$\exists^{\log^2 n}\forall^{\log n}\exists^{\log n}\mathsf{DTISP}(\mathrm{polylog}(n),\log(n)) \subseteq \exists^{\log^2 n}\mathsf{L}\cap\exists^{\log^2 n}\mathsf{FOLL},$$

for QUASIGROUP ISOMORPHISM. In addition to improving upon the work of Chattopadhyay–Torán–Wagner [CTW13], the work of Collins–Grochow–Levet–Weiß yielded depth-$3\frac{1}{2}$ circuits of size $n^{O(\log n)}$. In contrast, our bound requires non-deterministically guessing $O(\log^c n)$ bits, for some constant $c > 2$ that we do not estimate carefully. Thus, even for GROUP ISOMORPHISM, although we improve the depth, it is at the cost of increasing the size.

| Year | Result | Depth | Citation |
|---|---|---|---|
| 1970 | Generator-enumerator introduced | $\mathrm{poly}(n)$ | Felsch & Neubüser [FN70][6] |
| 1978 | $\exists^{\log^2 n}\mathsf{P} \subseteq \mathsf{DTIME}(n^{\log n+O(1)})$ | $\mathrm{poly}(n)$ | Tarjan (see Miller [Mil78]) |
| 1977 | $\mathsf{DSPACE}(\log^2 n)$ | $O(\log^2 n)$ | Lipton–Snyder–Zalcstein [LSZ77][7] |
| 1994 | $\exists^{\log^2 n}\mathsf{AC}^1$ | $O(\log n)$ | Wolf [Wol94][8] |
| 2010 | $\exists^{\log^2 n}\mathsf{SAC}^1$ | $O(\log n)$ | Wagner [Wag10] |
| 2010 | $\exists^{\log^2 n}\mathsf{FOLL}\cap\exists^{\log^2 n}\mathsf{L}$ | $O(\log\log n)$ | Chattopadhyay–Torán–Wagner [CTW13][9] |
| 2013 | $\exists^{\log^2 n}\mathsf{SC}^2\cap\exists^{\log^2 n}\mathsf{L}$ | $O(\log n)$ | Papakonstantinou–Tang–Qiao [Tan13][10] |
| 2024 | $\Sigma_3\mathsf{TISP}(\mathrm{polylog}(n),\log(n)) \subseteq \mathsf{quasiAC}^0$ | $3\frac{1}{2}$ | Collins–Grochow–Levet–Weiß [CGLW25] |
| 2026 | $\Sigma_2\mathsf{TIME}(\mathrm{polylog}(n)) \subseteq \mathsf{quasiAC}^0$ | $2\frac{1}{2}$ | This work |

Table 1: History of the low-level circuit complexity of algorithms for (QUASI)GROUP ISOMORPHISM based on the generator-enumerator technique, in comparison to this paper (all but the last row of the table are from [CGLW25, Table 1]). For non-circuit classes, we list their depth as the best-known depth of their simulation by circuits. The class in the bound obtained by Collins–Grochow–Levet–Weiß [CGLW25] is contained in all previous classes listed in the table, which also all apply to QUASIGROUP ISOMORPHISM. Compared to Collins–Grochow–Levet–Weiß, our work improves the circuit depth, but at the cost of increasing the size (in the exponent of the exponent of the quasipolynomial).

[6] Complexity not analyzed there, but the same as Tarjan's algorithm [Mil78].

[7] Despite the publication dates, this seems to have been independent of Tarjan's result. They note that Miller and Rabin had also observed this result independently.

[8] Wolf only claims a bound of $\exists^{\log^2 n}\mathsf{NC}^2$. However, if we replace his use of $\mathsf{NC}^1$ circuits to multiply two elements of a quasigroup with $\mathsf{AC}^0$ circuits, we immediately get the $\exists^{\log^2 n}\mathsf{AC}^1$ bound.

[9] They do not claim the $\exists^{\log^2 n}\mathsf{L}$ bound, but it follows immediately from their algorithm and results.

[10] We have written the result this way, despite $\exists^{\log^2 n}\mathsf{L} \subseteq \exists^{\log^2 n}\mathsf{SC}^2$, because in Tang's thesis [Tan13], the only place this is currently published, they only claim $\mathsf{NSC}^2$ using only $O(\log^2 n)$ bits of nondeterminism, which in our notation would be $\exists^{\log^2 n}\mathsf{SC}^2$. However, their algorithm and results also immediately yields a $\exists^{\log^2 n}\mathsf{L}$ bound.

**Short presentations.** See Section 2.4 for background on group presentations. We say that a presentation $P$ for a group $G$ is *short* if $P$ has length $\mathrm{polylog}(|G|)$. Babai and Szemerédi [BS84] first introduced the *Short Presentation Conjecture*, which asks whether there exists an absolute constant $c$, such that every finite simple group of order $n$ has length $\leq (\log n)^c$. Babai, Goodman, Kantor, Luks, and Pálfy [BGK+97] subsequently investigated the Short Presentation Conjecture. Following them, we say that presentations of finite simple groups are *uniform* if they can be constructed in $\mathsf{DTIME}(\mathrm{polylog}(n))$ from the standard names for the finite simple groups. We have mentioned in Section 1.2 the results on these conjectures most relevant for our paper [BS84, BGK+97, HS01, Suz62, HKSW20, GKKL08].

Prior to this work, the existing short presentations for $\mathrm{PSU}_3(q)$ and $\mathrm{Sz}(q)$ had not been shown to be uniform. We establish uniformity for both $\mathrm{PSU}_3(q)$ (Proposition 5.1) and $\mathrm{Sz}(q)$ (Proposition 6.1).

**Parallel algorithms for special cases of Group Isomorphism.** The work on $\mathsf{NC}$ algorithms for GpI and QGpI is comparatively nascent compared to that of GI (see [LRS24] for a survey of $\mathsf{NC}$ algorithms for GI). Indeed, much of the work involves parallelizing the generator-enumeration strategy (*ibid.*), which has yielded bounds of $\mathsf{L}$ [Tan13] for $O(1)$-generated groups and $\mathsf{SAC}^1$ for $O(1)$-generated quasigroups [Wag10]. Other families of groups known to admit $\mathsf{NC}$ isomorphism tests include Abelian groups [CTW13, GL26, CGLW25], graphical groups arising from the CFI graphs [BS20, CL24, Col23], coprime extensions $H \ltimes N$ where $H$ is $O(1)$-generated and $N$ is Abelian [GL26] (parallelizing a result from [QST11]), groups of almost all orders [CGLW25] (parallelizing [DW22]), and Fitting-free groups where the number of non-Abelian simple factors of the socle is $O(\log n/\log\log n)$ [GL26] (parallelizing a result from [BCGQ11]). Grochow, Johnson, and Levet recently exhibited an $\mathsf{AC}^3$ isomorphism test for the class of all Fitting-free groups [GJL25].

Johnson, Levet, Vojtěchovský, and Widholm [JLVW26] recently showed that in the multiplication table model, a fully-refined direct product decomposition of a group can be computed in $\mathsf{AC}^3$. They also exhibited an $\mathsf{NC}$ isomorphism test for central quasigroups. To the best of our knowledge, no other special family of quasigroups that are not groups (beyond $O(1)$-generated quasigroups) are known to admit even a polynomial-time isomorphism test.

Levet [Lev26] recently exhibited an $\mathsf{AC}^3$ isomorphism test for coprime extensions $H \ltimes N$ where $H$ is elementary Abelian and $N$ is Abelian (parallelizing the remainder of [QST11]). Additionally, Levet exhibited an $\mathsf{AC}^3$ isomorphism test for central-radical groups where $\mathrm{Rad}(G)$ was elementary Abelian; and $G/\mathrm{Rad}(G)$ was a direct product of either (i) non-Abelian simple groups, or (ii) perfect groups of bounded size (parallelizing a result of Grochow and Qiao [GQ17]).

**First-order encodings of groups: the work of Nies and Tent [NT17].** Instead of using a group presentation as its short encoding, one could use more general first-order sentences in the language of groups, in the vein of descriptive complexity. In that setting, Nies and Tent showed an analogue of Short Presentation Conjecture: that all groups have polylogarithmic-length first-order descriptions [NT17].

We observe here that one can directly get a $\mathsf{quasiAC}^0$ upper bound from their results; however, we will also discuss the obstacle we faced in getting their results to yield depth-3 circuits. Since it is not used elsewhere in our paper, we refer to their paper for any needed background and definitions.

A first-order sentence identifies a group $G$ if $G$ is the only group, up to isomorphism, that satisfies that sentence. Nies and Tent [NT17, Thm. 7.3] proved that there is a constant $m$ such that every finite group $G$ has a $\Sigma_m$-sentence $\varphi_G$ of length $O(\log^4 |G|)$ that identifies $G$. Hence, for

two finite groups $G, H$, we have the following:

$$G \ncong H \iff \exists \varphi \left[ |\varphi| \leq c \log^4 n,\ \varphi \in \Sigma_m,\ G \models \varphi,\ H \not\models \varphi \right],$$

where $n$ denotes the input size and $c$ is a suitable constant. Indeed, if $G \ncong H$, we may take $\varphi = \varphi_G$.

**Observation 1.5.** *Theorem 7.3 of Nies and Tent [NT17] implies that* Group Isomorphism *is in* $\mathsf{quasiAC}^0$.

*Proof.* For each fixed candidate sentence $\varphi$, the predicates $G \models \varphi$ and $H \models \varphi$ are computable by constant-depth quasipolynomial-size circuits via the standard quantifier-to-gate translation: each existential quantifier block becomes an $\mathsf{OR}$, each universal block becomes an $\mathsf{AND}$, and atomic formulas are evaluated in constant depth from the multiplication tables. Since $|\varphi| \leq O(\log^4 n)$ and the number of quantifier alternations is bounded by some constant $m$, this yields a $\mathsf{quasiAC}^0$ circuit for the inner test. Finally, the outer existential over $\varphi$ ranges over only $2^{(\log n)^{O(1)}}$ candidates, so it can be implemented by one additional $\mathsf{OR}$ gate of quasipolynomial fan-in, preserving $\mathsf{quasiAC}^0$. □

We note, however, that they estimate a bound of $m \leq 10$ (for most of their paper they can get $\Sigma_3$ sentences, but not for the Ree groups). However, even if they could get $m$ down to 3, we do not see how to get circuits better than depth $3\frac{1}{2}$ out of this approach: the outer existential to guess $\varphi$ can be merged with the top $\mathsf{OR}$ gate in evaluating $\varphi$, but then evaluating the atomic (unquantified) formulas at the "bottom" of $\varphi$ seems to require $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$, which increases the depth by one, to 4 ($3\frac{1}{2}$ because the bottom fan-in is only polylogarithmic).[11]

**Lower bounds for Group Isomorphism.** We are not aware of prior lower bounds on GpI on any class of Boolean circuits; nonetheless, there are several lower bounds in more restricted algebraic models of computation, which we review here. Gowers [Gow11], reiterated by Babai [Bab16, Page 81], asked whether there exists some fixed constant $k$, such that the profiles of subgroups generated by at most $k$ elements would suffice to place Group Isomorphism into $\mathsf{P}$. That is: are all groups determined up to isomorphism by the multiset of isomorphism types of their $k$-generated subgroups? Glauberman and Grabowski [GG15] refuted this claim, by showing that $k \geq \sqrt{2 \log_3 |G|} - 5/2$ was necessary. Wilson [Wil19] subsequently improved this bound to $k \geq \log_3 |G| - 2$. It is remarkable how tight this bound is: every group is generated by at most $\log_p |G|$ generators, where $p$ is the smallest prime dividing $|G|$.

Grochow and Levet [GL26] showed that there exists an infinite family of Abelian groups $(G_m)_{m \in \mathbb{N}}$ such that any $\mathsf{FO}$ (first-order) formula identifying $G_m$ requires $\Omega(\log |G_m|)$ variables. Collins and Levet [CL24] subsequently extended this result to a higher-arity analogue of the Ehrenfeucht–Fraïssé pebble game corresponding to $\mathsf{FO}$.

Bshouty [Bsh26] recently established a query lower bound of $\Omega(|G|)$ for isomorphism testing of Abelian groups given by their multiplication tables, in the "algebraic" model where queries are to group elements and operations are the natural group operations (not bit-wise).

The Abelian groups used for the aforementioned lower bounds in finite model theory [GL26, CL24] and query complexity [Bsh26] are essentially the same family of groups we use to get our CNF lower bound: products of many copies of the cyclic groups $\mathbb{Z}_2$ and $\mathbb{Z}_4$. However, for our CNF lower bound we need them in a particular form, in order to get a reduction from Matrix Rank.

[11] In comparison to [CGLW25], this would be depth $3\frac{1}{2}$ circuits of size $n^{O(\log^3 n)}$, whereas [CGLW25] get depth-$3\frac{1}{2}$ circuits of size $n^{O(\log n)}$.

## 2 Preliminaries

Throughout, for a finite group $G$ we write $n := |G|$ for its order. The symmetric group, consisting of permutations of a set of size $m$, is denoted $\mathrm{Sym}(m)$. $\mathrm{GL}(d, q)$ denotes the general linear group consisting of the $d \times d$ invertible matrices over the finite field $\mathbb{F}_q$. For matrix groups over finite fields, we write $q = p^e$, where $p$ is prime and $e \geq 1$, and we use $d$ for the matrix dimension, writing for example $\mathrm{GL}(d, q)$ and $\mathrm{SL}(d, q)$. Other auxiliary integer parameters will be denoted by letters such as $r, s, t, k$, and not by $n, d, p, q, e$, which we try to reserve for the meanings above.

### 2.1 Complexity theory

We assume that the reader is familiar with standard complexity classes such as $\mathsf{P}, \mathsf{NP}, \mathsf{L}$, and $\mathsf{NL}$. For a standard reference on circuit complexity, see [Vol99]. We consider Boolean circuits using the gates $\mathsf{AND}$, $\mathsf{OR}$, and $\mathsf{NOT}$.

In this paper, we will consider $\mathsf{DTIME}(\log^c n)$-uniform circuit families $(C_n)_{n\in\mathbb{N}}$, for some fixed $c \geq 1$. For this, one encodes the gates of each circuit $C_n$ by bit strings of length $O(\log^c n)$. Then the circuit family $(C_n)_{n\geq 0}$ is called $\mathsf{DTIME}(\log^c n)$-*uniform* if (i) there exists a deterministic Turing machine that computes for a given gate $u \in \{0,1\}^*$ of $C_n$ ($|u| \in O(\log^c n)$) in time $O(\log^c n)$ the type of gate $u$, where the types are $x_1, \ldots, x_n$, $\mathsf{NOT}$, $\mathsf{AND}$, or $\mathsf{OR}$ gates, and (ii) there exists a deterministic Turing machine that decides for two given gates $u, v \in \{0,1\}^*$ of $C_n$ ($|u|, |v| \in O(\log^c n)$) and a binary encoded integer $i$ with $O(\log^c n)$ many bits in time $O(\log^c n)$ whether $u$ is the $i$-th input gate for $v$. When $c = 1$, this notion of uniformity is referred to as $\mathsf{DLOGTIME}$-uniformity. For circuit families of size $\mathrm{poly}(n)$, we will use $\mathsf{DLOGTIME}$-uniformity.

**Definition 2.1.** Fix $k \geq 0$. We say that a language $L$ belongs to (uniform) $\mathsf{AC}^k$ if there exist a (uniform) family of circuits $(C_n)_{n\in\mathbb{N}}$ over the $\mathsf{AND}, \mathsf{OR}, \mathsf{NOT}$ basis such that the following hold:

- The $\mathsf{AND}$ and $\mathsf{OR}$ gates take exactly 2 inputs. That is, they have fan-in 2.
- $C_n$ has depth $O(\log^k n)$ and uses (has size) $n^{O(1)}$ gates. Here, the implicit constants in the circuit depth and size depend only on $L$.
- $x \in L$ if and only if $C_{|x|}(x) = 1$.

Note that the AND and OR gates can have arbitrary fan-in.

The complexity class $\mathsf{NC}^k$ is defined analogously as $\mathsf{AC}^k$, except that the $\mathsf{AND}, \mathsf{OR}$ gates are required to have fan-in at most 2. The class $\mathsf{SAC}^k$ is defined analogously, in which the $\mathsf{OR}$ gates have unbounded fan-in but the $\mathsf{AND}$ gates must have fan-in 2. We also allow circuits to compute functions by using multiple output gates.

For every $k$, the following containments are well-known:

$$\mathsf{NC}^k \subseteq \mathsf{SAC}^k \subseteq \mathsf{AC}^k \subseteq \mathsf{NC}^{k+1}.$$

In the case of $k = 0$, we have that:

$$\mathsf{NC}^0 \subsetneq \mathsf{AC}^0 \subsetneq \mathsf{NC}^1 \subseteq \mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{SAC}^1 \subseteq \mathsf{AC}^1.$$

We note that functions that are $\mathsf{NC}^0$-computable can only depend on a bounded number of input bits. Thus, $\mathsf{NC}^0$ is unable to compute the $\mathsf{AND}$ function. It is a classical result that $\mathsf{AC}^0$ is unable to compute Parity [FSS84, Ajt83, Yao85].

We will also be interested in NC and AC circuits of quasipolynomial size (i. e., $2^{O(\log^k n)}$ for some constant $k$). For a circuit class $\mathcal{C} \subseteq \mathsf{NC}$, the analogous class permitting a quasipolynomial number of gates is denoted $\mathsf{quasi}\mathcal{C}$. Note that DLOGTIME uniformity does not make sense for quasiNC, as we cannot encode gate indices using $O(\log n)$ bits. Instead, we will use $\mathsf{DTIME}(\mathrm{polylog}(n))$-uniformity for quasiNC [Bar92, FGSTT20].

**Bounded nondeterminism.** For a complexity class $\mathcal{C}$, we define $\exists^{\log^i n}\mathcal{C}$ to be the set of languages $L$ such that there exists an $L' \in \mathcal{C}$ such that $x \in L$ if and only if there exists $y$ of length at most $O(\log^i |x|)$ such that $(x, y) \in L'$. Similarly, define $\forall^{\log^i n}\mathcal{C}$ to be the set of languages $L$ such that there exists an $L' \in \mathcal{C}$ such that $x \in L$ if and only if for all $y$ of length at most $O(\log^i |x|)$, $(x, y) \in L'$. For any $i \geq 0$ and any $c \geq 0$, both $\exists^{\log^i n}\mathsf{FOLL}$ and $\forall^{\log^i n}\mathsf{FOLL}$ are contained in quasiFOLL, and so cannot compute Parity [CTW13, Smo87].

**Time- and space-restricted Turing machines.** When considering complexity classes defined by Turing machines with a time bound $t(n) \in o(n)$, we use Turing machines with random access to the input tape, which is read-only, and a separate address (or *index*) tape. After writing an address, the machine can go to a query state reading the symbol from the input at the location specified by the binary-encoded integer on the address tape. As usual, the machines are otherwise allowed to have multiple work tapes.

For functions $t(n), s(n) \in \Omega(\log n)$, the class $\mathsf{DTISP}(t(n), s(n))$ is defined to consist of decision problems computable by deterministic $t(n)$-time and $s(n)$-space-bounded Turing machines. Be aware that there must be one Turing machine that simultaneously satisfies the time and space bound. For details we refer to [Vol99, Section 2.6]. For further reading on the connection to $\mathsf{quasiAC}^0$, we refer to [Bar92, FGSTT20].

We frequently use $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$ and $\mathsf{DTIME}(\mathrm{polylog}(n))$. However, because of how small the time and space bounds are for these classes, when we abuse notation to say some function (not necessarily decision problem) is computable in $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$, there is some ambiguity as to what this might mean. We use the following two equivalent definitions. First we need some setup. For a function $f$ from bit-strings to bit-strings, we define its *bit function* bit-$f$ as follows. Write $f(x)_i$ to denote the $i$-th bit of $f(x)$. Then define

$$\text{bit-}f(x, i) = \begin{cases} f(x)_i & i \leq |f(x)| \\ \bot & i > |f(x)|. \end{cases}$$

Note that bit-$f$ is always a function which we may take to have at most two output bits (e. g., by encoding $f(x)_i$ by repeating the bit, and encoding $\bot$ by 01), i. e., a pair of decision problems.

**Definition/Lemma 2.2** ([CGLW25, Definition/Lemma 2.4])**.** *For any function $f\colon \{0,1\}^* \to \{0,1\}^*$ with $|f(x)| \leq O(\log |x|)$, the following are equivalent:*

1. *$f$ is computable by a Turing machine, using $\mathrm{poly}(\log n)$ time and $O(\log n)$ space, that halts with the result written in a specified place on its work tape.*
2. *The two bits of bit-$f$ are each decision problems in $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$, i. e., can each be decided by a Turing machine using $\mathrm{poly}(\log n)$ time and $O(\log n)$ space.*

*In either case, we say $f$ is computable in $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$.*

The same result and proof work up to $|f(x)| \leq \text{poly}(\log |x|)$, if we provide the machine with a separate output tape whose size is not counted towards the size bound.

**Fact 2.3** (cf. [CGLW25, Fact 2.5])**.** *For any $t(n) \leq n$,*[12] *any decision problem in $\mathsf{DTIME}(t(n))$ is computable by decision trees of depth $t(n)$, hence by CNFs with $2^{t(n)}$ clauses each of width $t(n)$, as well as by DNFs with $2^{t(n)}$ terms each of width $t(n)$.*

**Proposition 2.4.** *For $t(n) \geq \log n$, any decision problem in $\mathsf{DTIME}(t(n))$ is in $\exists^{O(t(n)^2)}\mathsf{AC}^0$.*

Rather than following the proof of [CGLW25, Fact 2.5], the proof here is more similar to the proof of the Cook–Levin Theorem. It seems plausible to us that a proof similar to that of [CGLW25, Fact 2.5] might replace the $O(t(n)^2)$ non-deterministic bits with only $O(t(n))$, but we did not need that level of granularity so have not worked out the details.

*Proof.* Suppose $M$ is a $\mathsf{DTIME}(t(n))$ machine. We may consider its computation history on input $x$ as a $t(n) \times t(n)$ array, with space indexing the first coordinate and time the second. In this array, for each time step of the computation we include the contents of its work tapes and its query tape, but not the input tape. Each cell in the array contains information about the value of that tape cell at that time, as well as whether the tape head was in that tape cell at that time, and what state $M$'s control was in. Note that the address tape need only have length $\log n$ regardless of how large $t(n)$ is, since the address is an index into the input, which has length $n$.

The $O(t(n)^2)$ nondeterministic bits are used to guess the contents of the entire array, and an $\mathsf{AC}^0$ machine then verifies the contents as follows. (The constant hidden by the big-Oh accounts for: the number of tapes, the number of states, and adding in the size of the address tape.) For cells in which the state of the machine is not querying the input, the verification proceeds exactly as in Cook–Levin: each $2 \times 3$ sub-array is verified independently in parallel (2 adjacent time steps of 3 adjacent tape cells).

For cells in which the state of the machine is querying the input, note that our $\mathsf{AC}^0$ verifier circuit $C$ gets access both to the original input $x$ and to the nondeterministic guesses (for the contents of the computation history array). Thus, when verifying a cell in which the machine was in a query state, the $\mathsf{AC}^0$ verifier includes a gadget that examines the $\log n$ bits of the address tape, and for each of the $n$ possible settings of those bits accesses the corresponding bit of $x$, and then uses that bit to verify that the next cells in the array are consistent with that bit of $x$ (in terms of how it affected the state of the machine, its position, and what it writes on the tape). To give a more concrete sense of this gadget, we show how it accesses the indexed bit of the input: let $\alpha$ be the $\log n$ bits on the address tape, and for an integer $i \in [0, n]$ let $b(i, j)$ be the $j$-th bit of $i$ when written in binary. Then the gadget uses

$$\bigvee_{i=0}^{n-1} \left( x_i \wedge \bigwedge_{j=0}^{\lceil \log n \rceil} [b(i,j) = \alpha_j] \right).$$

This gadget thus has size $O(n \log n)$ and depth 2 (note that $b(i, j)$ is a constant, so $[b(i, j) = \alpha_j]$ is really just a literal: $\alpha_j$ if $b(i, j) = 1$ and $\neg\alpha_j$ if $b(i, j) = 0$).

The overall circuit thus has size $O(t(n)^2 n \log n)$, which is polynomial in the size of its input plus the number of non-deterministic bits. □

[12] Both Fact 2.3 and Proposition 2.4 continue to hold when $t(n) > n$, but tighter statements are available in that setting. In this paper we will only use this in the setting where $t(n) \leq \text{polylog}(n)$, so we do not bother specifying this in detail.

We will now recall the polylogarithmic-time hierarchy.

**Definition 2.5.** For $k \in \mathbb{N}$, let $\Sigma_k\mathsf{TIME}(\text{polylog}(n))$ be the class of languages $L$, such that:

$$\omega \in L \iff \exists x_1 \forall x_2 \exists x_3 \cdots Q x_k \, \mathsf{DTIME}(\text{polylog}(|\omega|)).$$

Here, the quantifiers alternate starting with an existential quantifier. In particular, if $k$ is even, then $Q$ is a universal quantifier. Otherwise, $Q$ is an existential quantifier.

We similarly define $\Pi_k\mathsf{TIME}(\text{polylog}(n))$ to be the class of languages $L$ such that:

$$\omega \in L \iff \forall x_1 \exists x_2 \forall x_3 \cdots Q x_k \, \mathsf{DTIME}(\text{polylog}(|\omega|)).$$

Here, the quantifiers alternate starting with an universal quantifier. In particular, if $k$ is even, then $Q$ is an existential quantifier. Otherwise, $Q$ is a universal quantifier.

In both the settings of $\Sigma_k\mathsf{TIME}(\text{polylog}(n))$ and $\Pi_k\mathsf{TIME}(\text{polylog}(n))$, we note that the certificates $x_1, \ldots, x_k$ all may be taken to have length $\text{polylog}(n)$, so in particular we have:

$$\Sigma_k\mathsf{TIME}(\text{polylog}(n)) = \exists^{\text{polylog}(n)} \forall^{\text{polylog}(n)} \cdots Q^{\text{polylog}(n)} \mathsf{DTIME}(\text{polylog}(n)).$$

The *polylogarithmic-time hierarchy*, which we denote by $\mathsf{PLH}$, is:

$$\mathsf{PLH} := \bigcup_{k \in \mathbb{N}} \Sigma_k\mathsf{TIME}(\text{polylog}(n)).$$

**Reductions.** A *many-one* reduction $A \leq_m B$ is a function $r$ such that for all strings $x$, $A(x) = B(r(x))$; the many-one reduction lies in a complexity class $\mathcal{C}$ if $r \in \mathcal{C}$. A many-one reduction $r$ is a *projection* if each output bit of $r(x)$ is either a constant or some bit $x_i$ of the input (each $x_i$ may be used zero or more times). We say the function $n \mapsto \max_{|x|=n} |r(x)|$ is the *stretch* of $r$. Note that if $r$ is a projection reduction of stretch $s(n)$ (of any complexity) and $B$ is solvable by non-uniform CNFs (resp., DNFs) of size $S(n)$, then $A$ is solvable by non-uniform CNFs (DNFs) of size $S(s(n))$. In particular, if $r$ has polynomial stretch, and $B$ has quasi-polynomial size CNFs, then so does $A$. If $r$ is uniformly computable, then the analogous statements holds for uniform CNFs (resp. DNFs).

We say that a many-one reduction $r$ is an $\mathbb{F}_2$-*affine projection* if, after identifying its input and output strings with vectors over $\mathbb{F}_2$, every output bit of $r(x)$ is an affine linear function of the input bits. This means that for every output coordinate $j$ there are $a_{j,0}, a_{j,1}, \ldots, a_{j,n} \in \mathbb{F}_2$ such that $r(x)_j = a_{j,0} + \sum_{i=1}^{n} a_{j,i} x_i \pmod 2$. Differently put, $r(x) = Ax + b$ for some matrix $A$ over $\mathbb{F}_2$ and some vector $b$. We call a many-one reduction whose reduction map has this form an $\mathbb{F}_2$-affine projection reduction.

## 2.2 Group theory

For a standard reference, see [Rob82]. All groups will be assumed to be finite, unless otherwise specified. For $g, h \in G$, the *commutator* $[g, h] := ghg^{-1}h^{-1}$. The *commutator subgroup* $[G, G] := \langle \{[g, h] : g, h \in G\} \rangle$.

A group $G$ is *simple* if the only normal subgroups are $1, G$. A *composition series* for the group $G$ is an ascending chain of subgroups $1 = G_0 \triangleleft G_1 \triangleleft \cdots \triangleleft G_k = G$ such that for each $0 \leq i < k$, $G_{i+1}/G_i$ is simple. The multiset $\{\!\{G_{i+1}/G_i : 0 \leq i < k\cdot\}\!\}$ is the collection of *composition factors*. While a group may have many different composition series, the Jordan–Hölder Theorem states that

any two composition series of $G$ have the same length and same multiset of composition factors (up to isomorphism), though the composition factors need not appear in the same order between the two series.

The Classification of Finite Simple Groups says that every finite simple group is either alternating, of Lie type, or one of the 26 sporadic groups. Thus each finite simple group belongs to one of finitely many uniform families (each being indexed by one or two parameters: an integer $d$ and a prime power $q$), together with finitely many sporadic cases. Because of its utility in algorithms, we recall a convention from [BGK+97]:

**Definition 2.6** (Standard names of finite simple groups, [BGK+97])**.** The *standard name* of a finite simple group $G(q)$ includes: (1) a string indicating which family it is from (e.g. $A$, $B$, $C$, $D$, ${}^2G_2$, etc.), (2) if it is from a family of unbounded rank, the $d$ indexing which element of the family (viz. $A_d$, $B_d$, etc.), (3) the prime $p$ such that $q = p^e$, together with an irreducible polynomial over $\mathbb{F}_p$ defining the field occurring in the standard matrix realization of $G(q)$, and a primitive root of that field. For untwisted groups this field is $\mathbb{F}_q$; for twisted groups it may instead be $\mathbb{F}_{q^2}$ or $\mathbb{F}_{q^3}$. (The standard name of a sporadic group only includes part (1).)

The alternating groups of degree $m$ have order $m!/2$. The Lie-type finite simple groups with parameters $(d, q)$ have order $q^{\Theta(d^2)}$, and those without the parameter $d$ have order $q^{O(1)}$. The standard name of a finite simple group $S$ thus always has bit length at most $O(\log |S|)$.

Let $G$ be a group and let $S = \{g_1, \ldots, g_m\} \subseteq G$. A *straight-line program* (SLP) over $S$ is a finite sequence $h_1, \ldots, h_t$ of elements of $G$ such that each $h_i$ is one of the following: a generator $g_j \in S$; the inverse of an earlier term, $h_r^{-1}$ for some $r < i$; or the product of two earlier terms, $h_r h_s$ for some $r, s < i$. The *value* of the SLP is its final term $h_t$. We say that an element $g \in G$ has an SLP of length $t$ over $S$ if there is such a program with final value $g$, and we say a subset $T \subseteq G$ has an SLP of length $t$ over $S$ if there is such an SLP such that the set of elements appearing in the SLP contains $T$. Babai and Szemerédi [BS84] established a *Reachability Lemma*, showing that every element $h \in H := \langle S \rangle$ can be generated by an SLP of length at most $(1 + \log |H|)^2$.

We record here for use later a lemma whose proof uses the same idea as [CGLW25, Cor. 3.2]. Given a binary relation $R \subseteq X \times W$, we say that $R$ is computable in $\mathsf{coNTISP}(\mathrm{polylog}(n), \log n)\mathsf{MV}$ (cf. $\mathsf{NPMV}$, [BLS84, Sel94]) if there is a co-nondeterministic Turing machine using time $\mathrm{polylog}(n)$ and space $O(\log n)$ that, on input $x \in X$, each co-nondeterministic branch either outputs a special symbol †, or outputs some $w \in W$ such that $(x, w) \in R$, and furthermore, if there exists $w \in W$ such that $(x, w) \in R$, then at least one co-nondeterministic branch outputs some such $w$.

**Lemma 2.7** (Discrete logarithm in groups given by Cayley table)**.** *Given the multiplication table of a group $G$, discrete logarithms in $G$ can be computed in* $\mathsf{coNTISP}(\mathrm{polylog}(n), \log n)\mathsf{MV}$*. (That is, the binary relation $\{((G, g, h), k) : g, h \in G, g^k = h\}$ is computable in the stated complexity class.)*

*Proof (cf. [CGLW25, Cor. 3.2]).* If $h = 1$, simply output $k = 0$. Otherwise, universally check all $k = 1, \ldots, n-1$. We can check in $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$ whether $g^k = h$ by [CGLW25, Lem. 3.1]. If $g^k \neq h$, then that co-nondeterministic branch outputs †. If $g^k = h$, then that co-nondeterministic branch outputs $k$.

If $h \in \langle g \rangle$, then since $|\langle g \rangle| \leq |G| = n$, there must be some $k \in \{1, \ldots, n-1\}$ such that $g^k = h$, and hence at least one co-nondeterministic branch of the computation outputs an integer. □

## 2.3 Group Extensions and Cohomology

We recall background on group extensions and non-Abelian cohomology. Much of the theory was originally developed in [Ded64, Ina97], but [GQ17] independently redeveloped it (unaware of that prior work at the time), and put it in a form particularly convenient for algorithmic applications, so we follow the notation and development there. Given a finite group $G$, we will consider a normal subgroup $N \trianglelefteq G$ when considering $G$ as an extension of $N$ by $Q := G/N$. Here, we denote this as $N \overset{\iota}{\hookrightarrow} G \overset{\pi}{\twoheadrightarrow} Q$, where $\iota : N \to G$ is an injection and $\mathrm{Im}(\iota) = \ker(\pi)$. We refer to $G$ as the *total group* of the extension.

**Actions.** For $g \in G$, let $c_g : N \to N$ be given by $c_g(n) = gng^{-1}$. The conjugation action $\theta' : G \to \mathrm{Aut}(N)$ is defined by $\theta'(g) := c_g$. When $N$ is non-Abelian (as will frequently be the case in our setting), $\theta'$ does not contain $N$ in its kernel. However, the action of $N$ on itself by conjugation is by inner automorphisms (by definition); we denote the group of inner automorphisms by $\mathrm{Inn}(N)$. Thus, we obtain a well-defined homomorphism $G/N \to \mathrm{Aut}(N)/\mathrm{Inn}(N) =: \mathrm{Out(N)}$. We refer to such a map as an *outer action*.

In computations, rather than represent an outer action as a coset of $\mathrm{Inn}(N)$ in $\mathrm{Aut}(N)$, we specify it by a representative automorphism. Consequently, we must remember that we may need to multiply by an arbitrary element of $\mathrm{Inn}(N)$. We will use $T$ to denote an action rather than $\theta$, to remind the reader that the essential object is the outer action $\theta$ represented by $T$, despite the fact that we work with $T : Q \to \mathrm{Aut}(N)$. Two actions $T_1, T_2 : Q \to \mathrm{Aut}(N)$ are *outer equivalent* if there exists a function $t' : Q \to \mathrm{Inn}(N)$ and an automorphism $\alpha \in \mathrm{Aut}(N)$ such that $T_1(q) = \alpha^{-1} \circ t'(q) \circ T_2(q) \circ \alpha$, for all $q \in Q$.

**Definition 2.8** (Outer Action Compatibility [GQ17, Definition 3.2])**.** Given two actions $T_1, T_2 : Q \to \mathrm{Aut}(N)$, decide whether there exists $\beta \in \mathrm{Aut}(Q)$ such that $T_1$ and $T_2 \circ \beta$ are outer equivalent; that is, whether there exists $(\beta, \alpha, t') \in \mathrm{Aut}(Q) \times (\mathrm{Aut}(N) \rtimes \mathrm{Inn}(N)^Q)$ such that $T_1(q) = \alpha^{-1} \circ t'(\beta(q)) \circ T_2(\beta(q)) \circ \alpha$, for all $q \in Q$.

**Cohomology.** Let $\pi : G \to G/N \cong Q$ be the natural projection map. Any function $s : Q \to G$ such that $\pi(s(q)) = q$ for all $q \in Q$ is called a *section* of $\pi$. Any such section gives rise to a function $f_s : Q \times Q \to N$ defined by $f_s(p, q) = s(p)s(q) \cdot s(pq)^{-1}$. We are free to choose $s(1) = \mathrm{id}_G$, and then $f(1_Q, q) = f(q, 1_Q) = 1_N$ for all $q \in Q$. Such sections are called *normalized*. We will assume that all sections are normalized, unless otherwise stated.

A section gives rise to an action (not just an outer action) $T_s : Q \to \mathrm{Aut}(N)$, given by $T_s(q)(n) = s(q) \cdot n \cdot s(q)^{-1}$. The 2-cocycle identity is:

$$f_s(q_1, q_2) \cdot f_s(q_1 q_2, q_3) = T_s(q_1)(f_s(q_2, q_3)) \cdot f_s(q_1, q_2 q_3).$$

Any function $f : Q \times Q \to N$ is called a 2-*cochain*. If $f$ satisfies the 2-cocycle identity with respect to $T$, then $f$ is called a 2-*cocycle* with respect to $T$.

Observe that the 2-cocycle identity depends not only on the action $T_s$ and $f_s$, but also on the relationship between $T_s$ and $f_s$ (namely, that they come from the same section). It is preferable to have a condition that does not depend on the ambient group extension. To this end, note that the action satisfies $T_s(q_1)T_s(q_2) = c_{f_s(q_1,q_2)}(T_s(q_1 q_2))$, where $c_m : N \to N$ denotes the conjugation action sending $c_m(n) = mnm^{-1}$. We now recall the notion of extension data from Grochow and Qiao.

**Definition 2.9** (Extension Data [GQ17, Definition 3.4])**.** Let $Q$ and $N$ be groups. We say that a pair $(T, f)$ of an action $T : Q \to \mathrm{Aut}(N)$ and a function $f : Q \times Q \to N$ is *extension data* if for all $q_i \in Q$,

$$T(q_1)T(q_2) = c_{f(q_1,q_2)}(T(q_1q_2)), \text{ and}$$
$$f(q_1, q_2) \cdot f(q_1q_2, q_3) = T(q_1)(f(q_2, q_3)) \cdot f(q_1, q_2q_3).$$

In this case, we refer to $f$ as a 2-cocycle with respect to the action $T$.

We now recall what it means for two extension data to be equivalent.

**Definition 2.10** (Equivalence of Extension Data [GQ17, Definition 3.5])**.** Two extension data $(T_i, f_i)$ $(i = 1, 2)$ are *equivalent* if there exists a map $t : Q \to N$ such that $T_1(q) = c_{t(q)}T_2(q)$ for all $q \in Q$, and

$$f_1(q, r) = t(q)[T_2(q)(t(r))] \cdot c_{f_2(q,r)}(t(qr)^{-1}) \cdot f_2(q, r) := f_2^{t,T_2}(q, r).$$

**Definition 2.11** (Pseudo-Congruence of Extension Data [GQ17, Definition 3.6])**.** Let $Q$ and $N$ be groups. For $i = 1, 2$, let $T_i : Q \to \mathrm{Aut}(N)$, $f_i : Q \times Q \to N$, and $(T_i, f_i)$ the corresponding extension data. We say that $(T_1, f_1)$ and $(T_2, f_2)$ are *pseudo-congruent* if there exist $(\alpha, \beta) \in \mathrm{Aut}(N) \times \mathrm{Aut}(Q)$ and $t : Q \to N$ such that for all $q \in Q$ and all $n \in N$, the following hold:

$$T_1(q)(n) = (\alpha^{-1} \circ c_{t(\beta(q))} \circ T_2(\beta(q)) \circ \alpha)(n), \text{ and for all } q_1, q_2 \in Q$$
$$f_1(q_1, q_2) = \alpha^{-1}\left[f_2^{t,T_2}(\beta(q_1), \beta(q_2))\right] =: f_2^{(\alpha,\beta,t,T_2)}(q_1, q_2).$$

We now turn to discussing when two different extension data $(T_1, f_1), (T_2, f_2)$ yield isomorphic total groups. The following is essentially equivalent to [GQ17, Main Lemma 3.7], but using the assumption about isomorphisms extending from a given normal subgroup (which, in our setting, we will guess non-deterministically), rather than assuming the normal subgroups come from characteristic subgroup functions as in [GQ17]; the latter assumption is used in [GQ17] only to ensure that any isomorphism $G_1 \to G_2$ must send $N_1$ to $N_2$.

**Lemma 2.12** (cf. [GQ17, Main Lemma 3.7])**.** *Given two finite groups $G_1$ and $G_2$, with normal subgroups $N_1 \trianglelefteq G_1$ and $N_2 \trianglelefteq G_2$, we have that there is an isomorphism $G_1 \to G_2$ sending $N_1$ to $N_2$ if and only if both of the following conditions hold:*

*(a) $N_1 \cong N_2$ and $G_1/N_1 \cong G_2/N_2$.*

*(b) Let $(T_i, f_i)$ be extension data for $N_i$ and $G_i/N_i$. The extension data $(T_1, f_1)$ is pseudo-congruent to $(T_2, f_2)$.*

**Remark 2.13.** Following the introduction of $t\colon Q \to N$ starting on the bottom of p. 1168 of [GQ17], we see that the role of $t$ is essentially to correct for having chosen sections in two groups that are not equal under the chosen candidate isomorphism. In particular, Lemma 2.12 holds even if we restrict the pseudo-congruence in part (2) to have $t$ being the trivial map ($t(q) = 1$ for all $q \in Q$). That is, $G_1 \cong G_2$ by an isomorphism sending $N_1$ to $N_2$ iff (a) [as in Lemma 2.12] and (b') there exist extension data $(T_i, f_i)$ for $G_i$ $(i = 1, 2)$ that are pseudo-congruent by a pseudo-congruence with $t = 1$.

## 2.4 Group Presentations

We recall some preliminaries regarding group presentations. For a standard reference see, e. g., [CM80]; our presentation follows closely that in [DW22]. The *free group* $F[X]$ on a given alphabet $X$ is obtained by creating a disjoint copy $X^-$ of the alphabet, and treating the elements of $F[X]$ as words over the disjoint union $X \dot\cup X^-$, including the empty word $1 \notin X \cup X^-$. Formally, one can replace the given set $X$ by $\{(x, 1) : x \in X\}$ and $X^-$ by $\{(x, -1) : x \in X\}$. For simplicity, we abuse notation by identifying $x$ with $(x, 1)$, and $x^-$ with $(x, -1)$. The empty word serves as the identity, and concatenation is the group product. To impose the existence of inverses, we apply the rewriting rules $xx^- \mapsto 1$ and $x^-x \mapsto 1$, for each $x \in X$. For a set $M$, let $M^X$ denote the set of functions from $X$ to $M$. The elements of $M^X$ are naturally represented as tuples $\mathbf{m} = (\mathbf{m}_x)_{x\in X}$. For a group $G$, a tuple $\mathbf{g} \in G^X$, and a word $w \in F[X]$, we assign $w(\mathbf{g}) \in G$ by replacing each variable $x^\pm$ in $w$ with the corresponding value of $\mathbf{g}_x \in G$ or $\mathbf{g}_x^{-1} \in G$, and then evaluating the corresponding product in $G$. Define the homomorphism $\widehat{\mathbf{g}} : F[X] \to G$ by sending $w \mapsto w(\mathbf{g})$. If $G$ is generated by $S := \mathrm{Im}(\widehat{\mathbf{g}})$ and $R$ generates $\ker(\widehat{\mathbf{g}})$, then the pair $\langle S|R\rangle$ is a *presentation* of $G$, where $R$ is the set of *relations* relative to $S$. Note that $\langle S|R\rangle$ describes $G$ up to isomorphism, in which case we write $G \cong \langle S|R\rangle$.

In this paper, we will follow the conventions of [GKKL08] in defining the length of a presentation.

**Definition 2.14** (Length of a Presentation)**.** The *(word) length* of a presentation is the sum of the number of generators and the lengths of the relations as words in both the generators and their inverses.

**Remark 2.15** (On different notions of length)**.** There are differing notions in the literature regarding the length of a presentation. Babai, Goodman, Kantor, Luks, and Pálfy [BGK+97] take the length of the presentation to be the total number of characters required to write down all of the generators and relations. While Babai, Goodman, Kantor, Luks, and Pálfy count each generator as a single symbol, they point out that in properly accounting for the total bit-length, one could instead count each generator as an indexed symbol where the index must be written as a word over a fixed finite alphabet (say, binary). If there are $k$ generators, then doing so requires $O(\log k)$ bits per generator. As $k \le \lceil \log |G| \rceil$, this would only increase the length of the presentation by a factor of $O(\log k) = O(\log \log |G|)$.

In [BGK+97] they also allow exponents in their relators to be written in binary, but point out that one can also avoid this without substantially changing the total length, as follows. Namely, one can obtain a new presentation of the same asymptotic length by introducing new variables and relations to simulate repeated squaring. For instance, the cyclic group of order $m$ has presentation $\langle x|x^m = 1\rangle$, which has word length $m$, but bit-length $O(\log m)$. However, we can instead use the following presentation:

$$\langle x_0, \ldots, x_k | x_{i+1} = x_i^2 (0 \le i < k), x_0^{e_0} x_1^{e_1} \cdots x_k^{e_k} = 1\rangle,$$

where $e_i \in \{0, 1\}$ (for each $0 \le i \le k$) and $m = \sum_{i=0}^{k} 2^i e_i$, which has word length at most $4 \log(m) + O(1)$ [BGK+97, Remark 1.3].

Putting these all together, if a group $G$ has presentation of word length (Definition 2.14) $O(\log^c |G|)$ then it has a presentation of bit length $O(\log^c |G| \log \log |G|) \subseteq \widetilde{O}(\log^c |G|)$, which is smaller than $O(\log^{c+\varepsilon} |G|)$ for all fixed $\varepsilon > 0$. If the number of generators was initially $O(1)$ and the presentation's word length was $\Omega(\log |G|)$, then the factor of $\log \log |G|$ is unnecessary.

Babai, Goodman, Kantor, Luks, and Pálfy [BGK+97, Theorem 8.3] showed that if every finite simple group admits a presentation of length $O(\log^c n)$, then every finite group admits a presentation of length $O(\log^{c+1} n)$. Their proof was constructive. We recall their construction here as we will use it in our main result.

Following their terminology, we call a set $S$ of generators of a group $G$ *efficient* if every element of $G$ can be written as a straight-line program over $S$ of length at most $2\log_2 |G|$. Babai and Szemerédi [BS84] showed that from any generating set for a group $G$, there is a SLP of length at most $\log^2 |G|$ that computes an efficient generating set of $G$ (see [BGK+97, Lemma 8.2]). (We follow much, but not all, of the notation there for ease of reference; one notable exception is that our indices increase—if $i < j$ then $G_i \leq G_j$—whereas theirs decrease.)

**Construction 2.16** ([BGK+97, Proof of Theorem 8.3])**.** Suppose there is $c \geq 2$ such that all finite simple groups of order $n$ have presentations of length $O(\log^c n)$. Let $G$ be a finite group with composition series $1 = G_0 \trianglelefteq G_1 \trianglelefteq \cdots \trianglelefteq G_m = G$. We will describe their construction of a presentation for $G$ of length $O(\log^{c+1} |G|)$.

For each $i$, let $\langle \Gamma_i | R_i \rangle$ be the presentation for $G_i/G_{i-1}$ of length $O(\log^c |G_i/G_{i-1}|)$. Given a surjective homomorphism $\pi\colon X \to Y$, we say that $x \in X$ is a *lift* of $y \in Y$ if $\pi(x) = y$, and we say that $\widehat{S} \subseteq X$ is a lift of $S \subseteq Y$ if $|\widehat{S}| = |S|$ and $\pi(\widehat{S}) = S$.

The generators in the presentation of $G$ will be as follows:

(Gen1) For $i = 1, \ldots, m$, let $\Gamma_i^0$ be a minimum-size subset of $\Gamma_i$ that generates $G_i/G_{i-1}$ (in particular, $|\Gamma_i^0| \leq \log |G_i/G_{i-1}|$). Let $\widehat{\Gamma}_i$ be lifts of $\Gamma_i$ in $G_i$ (hence in $G$), along the natural quotient map $G_i \to G_i/G_{i-1}$. Let $\widehat{\Gamma}_i^0$ be the subset of $\widehat{\Gamma}_i$ that is a lift of $\Gamma_i^0$. Let $S_i$ be a short SLP computing an efficient set of generators of $G_i$ from $\Gamma_1^0 \cup \Gamma_2^0 \cup \cdots \cup \Gamma_i^0$, and let

$$S := \bigcup_{i=1}^{m} S_i.$$

(Gen2) For each $i$ and for each $\gamma \in \widehat{\Gamma}_i$, let $D(\gamma)$ be a an SLP of length $O(\log |G_i|)$ (since $S_i$ is an efficient generating set) computing $\gamma$ from $S_i$. Let

$$D := \bigcup_{i=1}^{m} \bigcup_{\gamma \in \widehat{\Gamma}_i} D(\gamma).$$

(Gen3) For each $i$ and each $\rho \in R_i$ giving the relation $\rho(\gamma_{i1}, \ldots, \gamma_{ir}) = 1$ in the generators $\Gamma_i = \{\gamma_{i1}, \ldots, \gamma_{ir}\}$, let $\widehat{\gamma}_{im} \in \widehat{\Gamma}_i$ be the lift of $\gamma_{im}$ chosen above. When we apply $\rho$ to the lifted elements, we find $\rho(\widehat{\gamma}_{i1}, \ldots, \widehat{\gamma}_{ir}) = z_\rho$ for some element $z_\rho \in G_{i-1}$. Let $P(\rho)$ be a short straight-line program computing $z_\rho$ from $S_{i-1}$. Let:

$$P := \bigcup_{i=1}^{m} \bigcup_{\rho \in R_i} P(\rho).$$

(Gen4) As $G_{i-1}$ is normal in $G_i$, we have for each $n \in G_{i-1}$ and each $g \in G_i$, that $gng^{-1} \in G_{i-1}$. For each $n \in \widehat{\Gamma}_{i-1}^0$ and $g \in \widehat{\Gamma}_i^0$, let $C(n, g)$ be a SLP of length $O(\log |G_i|)$ computing $gng^{-1}$ from $S_{i-1}$. Let:

$$C := \bigcup_{i=1}^{m} \bigcup_{n \in \widehat{\Gamma}_{i-1}^0} \bigcup_{g \in \widehat{\Gamma}_i^0} C(n, g).$$

Let $T := S \cup P \cup D \cup C$ be the set of generators of $G$ ($S$ alone generates $G$, but the additional generators will enable a shorter presentation).

In order to construct a presentation of $G$, they associate for each $t \in T$, an abstract generator symbol $x(t)$. Relative to this generating set, the following relations define $G$:

(Rel1) If some $t \in T$ arose as $t = uw$ or $t = w^{-1}$ in the course of one of the SLPs in $D$, $P$, or $C$, then include the relation $x(t) = x(u)x(w)$ or $x(t)x(w) = 1$, respectively.

(Rel2) For each relator $\rho \in R_i$ lifted to $\widehat{\rho}(\widehat{s}_{i1}, \ldots, \widehat{s}_{ir}) = z_\rho$, include the relation $\widehat{\rho}(x(\widehat{s}_{i1}), \ldots, x(\widehat{s}_{im})) = x(z_\rho)$.

(Rel3) For each $n \in \widehat{\Gamma}^0_{i-1}$ and $g \in \widehat{\Gamma}^0_i$, include the relation $x(gng^{-1}) = x(g)^{-1}x(n)x(g)$.

This completes the construction, whose correctness is proved in [BGK+97, Theorem 8.3].

### 2.5 Additional preliminaries for the Ree groups ${}^2G_2(q)$

We recall the specific facts about the Ree groups that will be used below. For the reader wishing to better understand these facts and put them in the larger context of finite simple groups, and Lie and algebraic groups, we refer to [Cur65, Tit74, Car89], but we will not need such depth here.

The Ree groups ${}^2G_2(q)$ [Ree60][13] are finite simple groups of order $q^3(q^3+1)(q-1)$, where $q = 3^{2k+1}$ is an odd power of 3. Rather than give an abstract definition, we use the $7 \times 7$ matrix representation of ${}^2G_2(q)$ from [KLM01], in the convenient form presented in [Bää14, Sec. 3].

Let $\ell = 3^k$; this parameter will show up in exponents frequently in this construction.[14] Then ${}^2G_2(q)$ is generated by

$$h(\lambda) := \mathrm{diag}(\lambda^\ell, \lambda^{1-\ell}, \lambda^{2\ell-1}, 1, \lambda^{1-2\ell}, \lambda^{\ell-1}, \lambda^{-\ell}) \qquad (\lambda \in \mathbb{F}_q^\times)$$
$$\Upsilon := \mathrm{antidiag}(-1,-1,-1,-1,-1,-1,-1).$$

along with a Sylow 3-subgroup of ${}^2G_2(q)$, which, when viewed as a subgroup of $\mathrm{GL}(7,q)$, is the intersection of ${}^2G_2(q)$ with the upper unitriangular matrices, and is generated by, for all $x \in \mathbb{F}_q$:

$$\alpha(x) = \begin{pmatrix} 1 & x^\ell & 0 & 0 & -x^{3\ell+1} & -x^{3\ell+2} & x^{4\ell+2} \\ 0 & 1 & x & x^{\ell+1} & -x^{2\ell+1} & 0 & -x^{3\ell+2} \\ 0 & 0 & 1 & x^\ell & -x^{2\ell} & 0 & x^{3\ell+1} \\ 0 & 0 & 0 & 1 & x^\ell & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -x & x^{\ell+1} \\ 0 & 0 & 0 & 0 & 0 & 1 & -x^\ell \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

[13] Sometimes called "small Ree groups" to distinguish them from the other groups discovered by Ree, ${}^2F_4(q)$, which have higher rank as algebraic groups. We avoid calling them "small" here because that word might be mistaken for other uses of "small" in our complexity-theoretic context, such as "polylogarithmic." Throughout this paper, when we say "Ree group", we mean the "small Ree groups ${}^2G_2(q)$."

[14] For context / to whet the reader's appetite to learn more: the so-called Frobenius automorphism of $\mathbb{F}_q$ is given by $x \mapsto x^3$; as an automorphism this has order $2k+1 = \log_3 q$. The function $x \mapsto x^\ell$ is the $k$-th power of the Frobenius automorphism, and has the property that its square is equal to the inverse of the Frobenius: $((x^\ell)^\ell)^3 = x^{3^{2k+1}} = x$. This is relevant to how ${}^2G_2(q)$ is constructed from the exceptional algebraic group $G_2(q)$.

$$\beta(x) = \begin{pmatrix} 1 & 0 & -x^\ell & 0 & -x & 0 & -x^{\ell+1} \\ 0 & 1 & 0 & x^\ell & 0 & -x^{2\ell} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 1 & 0 & x^\ell & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & x^\ell \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\gamma(x) = \begin{pmatrix} 1 & 0 & 0 & -x^\ell & 0 & -x & -x^{2\ell} \\ 0 & 1 & 0 & 0 & -x^\ell & 0 & x \\ 0 & 0 & 1 & 0 & 0 & x^\ell & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -x^\ell \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

This completes our definition/recollection of the Ree groups.

From the matrices above, one can directly verify the following facts (see also [War66] for more):

- As functions of $x$ and $y$, $\beta$ and $\gamma$ are "additive"; more precisely, for all $x, y \in \mathbb{F}_q$, we have $\beta(x+y) = \beta(x)\beta(y)$ and $\gamma(x+y) = \gamma(x)\gamma(y)$.
- $\langle \beta(x), \gamma(y) : x, y \in \mathbb{F}_q \rangle \cong \mathbb{F}_q^2 \cong \mathbb{F}_3^{4k+2}$ is an elementary Abelian group, which we denote $[U, U]$.
- For all $x, y \in \mathbb{F}_q$, there exists elements $b_{x,y} \in [U, U]$ such that $\alpha(x+y) = \alpha(x)\alpha(y)b_{x,y}$.

There are a few additional key structural facts about the group that we will recall and use. Let $U$ be the Sylow 3-subgroup $U := \langle \alpha(x), \beta(y), \gamma(z) : x, y, z \in \mathbb{F}_q \rangle$; let $H = \{h(\lambda) : \lambda \in \mathbb{F}_q^\times\} = \langle h(\omega) \rangle$, where $\omega$ is a generator of the multiplicative group $\mathbb{F}_q^\times$ (a.k.a. a primitive element of $\mathbb{F}_q$). We recall the following facts:

- $U = \{\alpha(a)\beta(b)\gamma(c) : a, b, c \in \mathbb{F}_q\}$ (that is, every element of $U$ can be represented as $\alpha(a)\beta(b)\gamma(c)$).
- (Bruhat decomposition) ${}^2G_2(q) = (HU) \sqcup (HU\Upsilon U)$, where we highlight that this is a *disjoint* union.
- $H$ normalizes $U$, and every element of $HU$ has a unique expression of the form $h(\lambda)\alpha(a)\beta(b)\gamma(c)$ with $\lambda \in \mathbb{F}_q^\times$ and $a, b, c \in \mathbb{F}_q$.

- Every element of ${}^2G_2(q)$ outside of $HU$ has a *unique* expression of the form $h(\lambda)u\Upsilon u'$ with $u, u' \in U$, and every element of the form $h(\lambda)u\Upsilon u'$ with $u, u' \in U$ is not in $HU$.

The above gives us a nice bijection or parametrization of the Ree groups:

$$\left(\mathbb{F}_q^\times \times \mathbb{F}_q^3\right) \sqcup \left(\mathbb{F}_q^\times \times \mathbb{F}_q^6\right) \to {}^2G_2(q), \tag{1}$$

given by

$$\mathbb{F}_q^\times \times \mathbb{F}_q^3 \ni (\lambda; a, b, c) \mapsto h(\lambda)\alpha(a)\beta(b)\gamma(c) \in HU$$

and

$$\mathbb{F}_q^\times \times \mathbb{F}_q^6 \ni (\lambda; a, b, c, x, y, z) \mapsto h(\lambda)\alpha(a)\beta(b)\gamma(c)\Upsilon\alpha(x)\beta(y)\gamma(z) \in HU\Upsilon U.$$

The facts recalled above immediately imply that this is a bijection.

In our proof, we will need an efficiently computable form of this bijection, relative to a particular generating set, which we give in Lemma 3.5.

# 3 Depth-3 Circuits for Group Isomorphism

In this section, we will establish the following theorem. We first introduce a small notation: for two classes of Boolean functions $\mathcal{C}, \mathcal{D}$, we use $\mathcal{C} \wedge \mathcal{D}$ to denote the class of Boolean functions expressible as a conjunction of a function in $\mathcal{C}$ and a function in $\mathcal{D}$.

**Theorem 3.1** (Full version of Theorem B)**.** *In the Cayley-table model,* Group Isomorphism *belongs to*

$$\exists^{\log^{c_1}(n)}\forall^{\log^{c_2}(n)}\mathsf{DTIME}(\mathrm{polylog}(n)) \wedge \forall^{\log n}\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$$

*for some constants $c_1, c_2$. Consequently,* Group Isomorphism *can be decided by uniform depth-3 circuits of quasipolynomial size and polylogarithmic bottom fan-in.*

*If furthermore, the Uniform Short Presentation Conjecture holds, then* Group Isomorphism *belongs to*

$$\exists^{\log^{c}(n)}\mathsf{DTIME}(\mathrm{polylog}(n)) \wedge \forall^{\log n}\mathsf{DTISP}(\mathrm{polylog}(n), \log(n)).$$

We will accomplish this in two steps. We will first show that we can verify whether a multiplication table satisfies the group axioms in $\forall^{\log n}\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$. We will show, more strongly, that this check can be implemented using a CNF of polynomial-size and logarithmic width (Lemma 3.3). Our second step will be to show that, given two multiplication tables of groups $G$ and $H$, we can decide whether $G$ and $H$ are isomorphic in $\exists^{\mathrm{polylog}(n)}\forall^{\mathrm{polylog}(n)}\mathsf{DTIME}(\mathrm{polylog}(n))$ (Proposition 3.4).

We note that as an immediate corollary, we show that the Uniform Short Presentation Conjecture implies a resolution of a question implicit in [CTW13] and highlighted in [CGLW25], up to the exponent of the exponent: Is GpI in $\exists^{\log^2 n}\mathsf{AC}^0$?

**Corollary 3.2.** *The Uniform Short Presentation Conjecture implies* GpI $\in \exists^{\mathrm{polylog}(n)}\mathsf{AC}^0$.

*Proof.* From the second half of Theorem 3.1, and using the polynomial-size CNF from Lemma 3.3, we get that the Uniform Short Presentation Conjecture implies that GpI is in

$$\exists^{\log^{c}(n)}\mathsf{DTIME}(\mathrm{polylog}(n)) \wedge \mathsf{CNF}.$$

By Proposition 2.4, we have $\mathsf{DTIME}(\mathrm{polylog}(n)) \subseteq \exists^{\mathrm{polylog}(n)}\mathsf{AC}^0$. Combining the two existential quantifiers and using the fact that $\mathsf{CNF} \subseteq \mathsf{AC}^0$, the result follows. □

### 3.1 The total language and validity of Cayley tables

It is more natural to talk about our circuits under the assumption or promise that the input Cayley tables are in fact the multiplication tables of *groups*. Here we show that this assumption can be checked by a polynomial-size CNF, so that later we can talk only in the language of groups, while still solving the total GPI problem rather than the promise problem.

**Lemma 3.3** (Validity of Cayley tables)**.** *The language* $\mathrm{VALID}_n$ *of* $n \times n$ *tables whose entries encode a group operation on* $[n]$ *is decidable in* $\forall^{\log n}\mathsf{DTISP}(\log n \log\log n, \log n)$*, as well as by a uniform CNF of polynomial size and logarithmic clause width.*

*Proof.* Assume $n \geq 2$. Let $\ell = \lceil \log_2 n \rceil$. The verifier checks that the table is an associative Latin square, i. e., it checks the following four universally quantified formulas testing: range, row-distinctness, column-distinctness, and associativity.

All universally quantified labels are encoded using $\ell$ bits. Since some $\ell$-bit strings may not represent elements of $[n]$, any universal instance involving such an invalid label is treated as vacuous. Actual table entries, however, are explicitly checked to lie in $[n]$. The conditions are as follows.

- First, every table entry is in range: for all $x, y \in [m]$, $T(x, y) \in [n]$.
- Second, every row has distinct entries: for all $x, y, z \in [n]$, if $y \neq z$ then $T(x, y) \neq T(x, z)$.
- Third, every column has distinct entries: for all $x, y, z \in [n]$, if $y \neq z$, then $T(y, x) \neq T(z, x)$.
- Finally, for all $x, y, z \in [n]$, associativity holds. Let $a := T(x, y)$, $b := T(y, z)$. The verifier checks $T(a, z) = T(x, b)$, that is, $T(T(x, y), z) = T(x, T(y, z))$.

In order to check these conditions, we will use $3\ell \in O(\log n)$ universally quantified (co-nondeterministic) bits. Given the co-nondeterministic bits, we may check each of the above conditions in $\mathsf{DTISP}(\log n \log\log n, \log n)$, where the main runtime comes from computing indices of the form $ni + j$ where $n, i, j$ are all $\log n$-bit numbers, which can be multiplied in time $O(\log n \log\log n)$ [HvdH21] and added in time linear in their bit-length. Thus, our total work is $\forall^{\log n}\mathsf{DTISP}(\log n \log\log n, \log n)$. By Fact 2.3, this can thus checked by a CNF of size $n^{O(\log\log n)}$. A more careful analysis yields a polynomial-size CNF. We will now proceed with this analysis.

In each case, we are only comparing table entries to the bound of $n$, or comparing two table entries to each other. A comparison between an $\ell$-bit table entry and a fixed $\ell$-bit number $n$ is expressed bit-by-bit. Thus, for fixed $c$, the condition $T(x, y) \neq c$ is a single clause of width $\ell$, while $T(x, y) = c$ is a conjunction of $\ell$ literals. Similarly, equality of two table entries, such as $T(a, z) = T(x, b)$, is a conjunction of $O(\ell)$ constant-size CNFs, namely the bitwise $\mathsf{XNOR}$s (equality).

The range condition is imposed by including, for each $x, y \in [n]$ and each $\ell$-bit string $c \notin [n]$, the clause $T(x, y) \neq c$. For the row- and column-distinctness checks, inequality of two table entries is imposed by forbidding each possible common value. Thus row-distinctness is expressed by the clauses $T(x, y) \neq c \ \vee \ T(x, z) \neq c$ for all $x, y, z, c \in [n]$ with $y \neq z$, and column-distinctness is expressed similarly by the clauses $T(y, x) \neq c \ \vee \ T(z, x) \neq c$. For associativity, we expand over the possible values $a, b \in [n]$ of $T(x, y)$ and $T(y, z)$. For each $x, y, z, a, b \in [n]$ and each bit position $r \in [\ell]$, we include the two clauses $T(x, y) \neq a \ \vee \ T(y, z) \neq b \ \vee \ \neg T(a, z)_r \ \vee \ T(x, b)_r$, and $T(x, y) \neq a \ \vee \ T(y, z) \neq b \ \vee \ T(a, z)_r \ \vee \ \neg T(x, b)_r$. These clauses say that if $T(x, y) = a$ and $T(y, z) = b$, then the $r$-th bits of $T(a, z)$ and $T(x, b)$ agree.

Altogether there are polynomially many clauses: $O(n^3)$ range clauses, $O(n^4)$ row-distinctness clauses, $O(n^4)$ column-distinctness clauses, and $O(n^5\ell)$ associativity clauses. Each clause has width $O(\ell)$. Hence the checks yield a uniform polynomial-size CNF.

These conditions characterize group tables. If $T$ is a group table, then all entries lie in $[n]$, every row and every column is a permutation of $[n]$, and associativity holds. Hence the verifier accepts. Conversely, suppose that the verifier accepts. The range condition ensures that $T$ defines a binary operation on $[n]$. The row-distinctness and column-distinctness conditions imply that every row and every column contains $n$ distinct elements of $[n]$. Thus every row and every column contains each element of $[n]$ exactly once, or equivalently, $T$ is a Latin square (the multiplication table of a quasigroup). Since the associativity check also holds and $[n]$ is nonempty, $T$ is a nonempty associative quasigroup. It is a standard fact that every nonempty associative quasigroup is a group, thus $T$ is the Cayley table of a group. Therefore the verifier accepts exactly the valid Cayley tables of groups. □

### 3.2 The promise language and testing for isomorphism in the second level of the Polylogarithmic-Time Hierarchy

In this section, we will establish the following.

**Proposition 3.4.** *Given two groups $G$ and $H$ by their multiplication tables, we can decide if $G \cong H$ in $\exists^{\mathrm{polylog}(n)}\forall^{\mathrm{polylog}(n)}\mathsf{DTIME}(\mathrm{polylog}(n))$. If furthermore, all of $G$'s composition factors admit uniform short presentations, then isomorphism between $G$ and $H$ can be decided in $\exists^{\mathrm{polylog}(n)}\mathsf{DTIME}(\mathrm{polylog}(n))$.*

Note that Proposition 3.4 operates under the promise that the multiplication tables provided as input indeed satisfy the group axioms. Because the details of the proof take 7+ pages, but the idea of the proof is much shorter, we start by presenting the idea:

*Proof idea.* We first present the proof idea for those composition factors that satisfy the Uniform Short Presentation Conjecture; then we present the additional ideas that we use to handle the Ree groups, for whom the Uniform Short Presentation Conjecture remains open (see "The status of the (Uniform) Short Presentation Conjecture" on p. 4 for more details).

**Proof idea assuming composition factors satisfy the Uniform Short Presentation Conjecture.** The existential quantifier is used to non-deterministically guess a presentation for $G$ that is of the form given in Construction 2.16, along with associated elements of $G$ that play the role of the generators in that presentation, and associated SLPs as in Construction 2.16. The verifier then checks that this presentation is satisfied by both $G$ and $H$. A priori this would just imply that $G$ and $H$ are both quotients of the presented group; however, the algorithm also checks that the orders of the simple composition factors it guesses—which can be computed using well-known formulas directly from the standard names—multiply out to $|G| = |H|$, and we can therefore conclude that the presented group has order exactly $n = |G| = |H|$, so if both $G$ and $H$ satisfy the presentation, then they are both isomorphic to the presented group, and hence to each other. The correctness essentially follows from [BGK+97, Theorem 8.3].

**Additional ideas to handle Ree group composition factors.** In general, we allow our non-deterministic guesses to guess that composition factors correspond to Ree groups as well, using

their standard name. However, we do not have available to us uniform short presentations of the Ree groups. Similar to the above case, we also use the non-deterministic bits to guess a generating set of size $O(\log n)$ (corresponding to a generating set we specify in Lemma 3.5). However, rather than verifying a cohomology class based on a small number of short relators, we instead use a co-nondeterministic $\forall^{\text{polylog}}$ quantifier to essentially treat the entire multiplication table as relators, that is, we look at relators of the form $q \cdot r \cdot (qr)^{-1}$. We use the universal quantifier both to enumerate over all pairs $(q, r)$—for which we take advantage of an easily computable and co-nondeterministically invertible bijection (see Lemma 3.5)—as well as, for each $(q, r)$ to pick an SLP in terms of the previous terms in the composition series.

There are two additional tricks we highlight here. The first is that the natural approach to the latter SLP would be to nondeterministically guess the SLP corresponding to $(q, r)$, but this would put us back in the third level of the polylogarithmic hierarchy. Instead, we try *all* SLPs of length $O(\log^2 n)$, with the bound being guaranteed by the Babai–Szemerédi Reachability Lemma. If the SLP is not equal to the element it is supposed to, then that co-nondeterministic branch simply accepts, essentially leaving the final accept/reject decision to the branches where the SLP *is* equal to what it is supposed to. And since we are guaranteed that such a branch exists, the check is still guaranteed to happen. Indeed, this is why we introduced the class $\mathsf{coNTISP}(\text{polylog}(n), \log n)\mathsf{MV}$.

The second is why we needed the efficiently invertible bijection of Lemma 3.5 at all. The issue here is the following: when we instead had a small number of relators, we can simply guess the SLPs—and hence the cohomology class—all at once and verify them. But when we are using the universal quantifier to quantify over relators, we have no way of knowing that our choices of section, and hence cohomology class, on the different co-nondeterministic branches were consistent with one another. But this consistency is precisely provided by the bijection of Lemma 3.5. The latter bijection also crucially takes advantage of another universal quantifier in order to solve discrete log efficiently in a group given by its multiplication table (Lemma 2.7).

With all these pieces in place, even for the Ree groups, the final verification checks are analogous to those for non-Ree groups coming from Construction 2.16 [BGK+97], but now the correctness instead relies on the non-Abelian cohomology setup of Lemma 2.12 [GQ17]. □

Before we come to the details of the proof of Proposition 3.4, we need one additional lemma about the Ree groups. For notation see Section 2.5. Suppose we are given $\mathbb{F}_q$ as $\mathbb{F}_3[x]/f(x)$. For $a \in \mathbb{F}_3^{2k+1}$, let us write

$$\overline{\alpha}^a := \alpha(x^0)^{a_0}\alpha(x^1)^{a_1}\cdots\alpha(x^{2k})^{a_{2k}},$$

and analogously for $\overline{\beta}^b$ and $\overline{\gamma}^c$. The following lemma gives us a "standard" generating set for ${}^2G_2(q)$ of size $2 + 3\log_3 q \leq O(\log |{}^2G_2(q)|)$, as well as normal forms for words for each element of the group, in terms of that generating set.

**Lemma 3.5.** *Suppose we are given an irreducible polynomial $f(x) \in \mathbb{F}_3[x]$ of degree $2k+1$ and a primitive element $\omega \in \mathbb{F}_q = \mathbb{F}_3[x]/f(x)$. Let*

$$B = \{h(\omega), \Upsilon\} \cup \{\alpha(x^i), \beta(x^i), \gamma(x^i) : 0 \leq i < 2k\}.$$

1. *The following function is a bijection:*

$$\varphi\colon ([q-1] \times \mathbb{F}_3^{3(2k+1)}) \sqcup ([q-1] \times \mathbb{F}_3^{6(2k+1)}) \to {}^2G_2(q)$$

*where $\varphi(i; a, b, c) = h(\omega)^i\overline{\alpha}^a\overline{\beta}^b\overline{\gamma}^c$ and $\varphi(i; a, b, c, a', b', c') = h(\omega)^i\overline{\alpha}^a\overline{\beta}^b\overline{\gamma}^c\Upsilon\overline{\alpha}^{a'}\overline{\beta}^{b'}\overline{\gamma}^{c'}$*

2. *The bijection $\varphi$ is computable in* $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$, *and*

$$\varphi^{-1} \in \mathsf{coNTISP}(\mathrm{polylog}(n), \log n)\mathsf{MV}.$$

*Proof.* 1. The bijection is nearly the same as that of (1). Using the facts recalled in the preliminaries that $\beta(y+z) = \beta(y)\beta(z)$ for all $y, z \in \mathbb{F}_q$ and similarly for $\gamma$, we have that $\overline{\beta}^b = \beta(b_0 + b_1 x + b_2 x^2 + \cdots + b_{2k} x^{2k})$, and similarly for $\gamma$. However, for $\alpha$, we only have that for all $y, z \in \mathbb{F}_q$, there exists $b_{y,z} \in [U, U]$ such that $\alpha(y+z) = \alpha(y)\alpha(z)b_{y,z}$. Thus we get that $\overline{\alpha}^a = \alpha(a_0 + a_1 x + a_2 x^2 + \cdots + a_{2k} x^{2k})\beta(f_1(a))\gamma(f_2(a))$ where $f_1, f_2 \colon \mathbb{F}_3^{2k+1} \to \mathbb{F}_q$ are the unique functions to make this equation hold, that is, $\beta(f_1(a))\gamma(f_2(a))$ is equal to the product of the appropriate $b_{y,z}$ terms that arise from rewriting $\overline{\alpha}^a$ as above.

Now, from the fact that (1) is a bijection, it will suffice to show that

$$\{\alpha(a)\beta(b)\gamma(c) : a, b, c \in \mathbb{F}_q\} = \{\overline{\alpha}^a \overline{\beta}^b \overline{\gamma}^c : a, b, c \in \mathbb{F}_3^{2k+1}\} \qquad \text{(to show)}.$$

Towards this end, let $\pi \colon \mathbb{F}_3^{2k+1} \to \mathbb{F}_q$ be the $\mathbb{F}_3$-linear vector space isomorphism given by $\pi(a) = a_0 + a_1 x + \cdots + a_{2k} x^{2k}$. Then from the preceding paragraph, we have

$$\overline{\alpha}^a \overline{\beta}^b \overline{\gamma}^c = \alpha(\pi(a))\beta(f_1(a))\gamma(f_2(a))\beta(\pi(b))\gamma(\pi(c)).$$

Since $[U, U]$ is Abelian and using the multiplicative property of $\beta, \gamma$, the latter is the same as

$$\alpha(\pi(a))\beta(f_1(a) + \pi(b))\gamma(f_2(a) + \pi(c)).$$

But now, as $b$ ranges over $\mathbb{F}_3^{2k+1}$, $\pi(b)$ bijectively ranges over $\mathbb{F}_q$. And since the additive group of $\mathbb{F}_q$ is, in particular, a group, $f_1(a) + \pi(b)$ bijectively ranges over $\mathbb{F}_q$ as well. Thus $\{\beta(f_1(a) + \pi(b)) : b \in \mathbb{F}_3^{2k+1}\} = \{\beta(\pi(b)) : b \in \mathbb{F}_3^{2k+1}\} = \{\beta(b) : b \in \mathbb{F}_q\}$, and similarly for $\gamma$. This proves that $\varphi$ is indeed a bijection.

2. That $\varphi$ is computable in $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$ is relatively straightforward: we may compute the $i$-th power of $h(\omega)$ by repeated squaring of $7 \times 7$ matrices, which takes only $O(\log i) \leq O(\log q) \leq O(\log |G|)$ many iterations, each of which involves $O(1)$ operations over $\mathbb{F}_q$. In total that uses $\mathrm{polylog}(q)$ time and $O(\log q)$ space. We may then construct and multiply the appropriate $\alpha, \beta, \gamma, \Upsilon$ matrices with similar complexity.

Computing $\varphi^{-1}(M)$ is slightly more involved. We split into two cases based on whether $M$ is upper-triangular or not.

- **Case 1:** Suppose first that $M$ is upper-triangular. Then $M = h(\omega)^i \alpha(a)\beta(b')\gamma(c')$ for some $i \in [q-1], a, b', c' \in \mathbb{F}_q$, and we must compute $i$ and $a, b, c \in \mathbb{F}_3^{2k+1}$ such that $M = h(\omega)^i \overline{\alpha}^a \overline{\beta}^b \overline{\gamma}^c$. In this case, $M_{1,1} = (\omega^i)^\ell$, where, following the notation of Section 2.5 we have $\ell = 3^k$. We may raise $M_{1,1}$ to the $3\ell$-th power by repeated squaring in $\mathbb{F}_q$ to get $\omega^i$. However, note that we do not yet have $i$, rather we have $\omega^i$ written as a polynomial in $x$. We use the co-nondeterministic bits to either output † or $i$ as in Lemma 2.7. On the branches of the discrete log subroutine that output †, our machine also outputs †; on the branches that output $i$, the machine continues as follows. First it computes $M' = h(\omega)^{-i} M$; then $M' = \alpha(a)\beta(b')\gamma(c')$ (but the algorithm does not yet know $a, b', c,'$). From the formulas for the matrices $\alpha(a), \beta(b'), \gamma(c')$ recalled in Section 2.5, one can see that $a = -M'_{3,6}$, that is, $a$ can simply be read off as an element of $\mathbb{F}_q$ from one of the entries of $M'$.

The algorithm then computes $M'' = (\overline{\alpha}^a)^{-1} M'$, which now has the form $\beta(b)\gamma(c)$ for some $b, c \in \mathbb{F}_q$ to be computed. Again, examining the formulas for the matrices, we find $b = -M''_{1,5}$, compute $M''' = \left(\overline{\beta}^b\right)^{-1} M'' = \beta(b)^{-1} M''$, and we find $c = -M'''_{2,7}$. This completes the computation of $\varphi^{-1}$ in the case that $M$ is upper triangular.

- **Case 2:** Finally, suppose instead that $M$ is not upper triangular. Then by facts recalled in Section 2.5, if $M$ is in ${}^2G_2(q)$ at all, it must be of the form $h(\omega)^i \alpha(a)\beta(b)\gamma(c)\Upsilon\alpha(a')\beta(b')\gamma(c')$ for appropriate $i \in [q-1]$ and $a, b, c, a', b', c' \in \mathbb{F}_q$. This time we proceed similarly (but we used Mathematica [WR26] to check our calculations symbolically, see Mathematica code in Appendix A). In this case, we have that $-(\omega^i)^{-\ell} = M_{7,1}$. To compute $\omega^{-i}$, we thus compute $(-M_{7,1})^{3\ell}$, and then to compute $i$ we use Lemma 2.7 as before.

  After left-multiplying by $h(\omega)^{-i}$ we find that $a^\ell$ is in the $(6,1)$ entry of the resulting matrix, so we can compute $a$ by taking the $3\ell$-th power of this entry. After left-multiplying by $(\overline{\alpha}^a)^{-1}$, we find that the $(3,1)$ entry is $-b$. Left multiplying by $\beta(b)^{-1}$, we can then find several more parameters at once: the $(2,1)$ entry is $-c$, the $(7,2)$ entry is $-(a')^\ell$. Finally, left multiplying by $(\gamma(c)\Upsilon\overline{\alpha}^{a'})^{-1}$, we are now in the situation where we have a matrix of the form $\beta(b')\gamma(c')$, and we can recover $b'$ and $c'$ as in the upper triangular case above.

Thus, every co-nondeterministic branch that does not output † indeed outputs $\varphi^{-1}(M)$, and at least one such co-nondeterministic branch outputs $\varphi^{-1}(M)$. □

We now flesh out the details of the above proof idea for Proposition 3.4:

*Proof of Proposition 3.4.* Our machine will non-deterministically guess polylog($|G|$)-many bits. When we encounter a composition factor of ${}^2G_2(q)$, we will also utilize polylog($|G|$)-many co-nondeterministic bits to handle isomorphism testing of the composition factor (using ideas from the $\Sigma_2\mathsf{TIME}(\text{polylog}(|G|))$ isomorphism test of Babai and Szemerédi [BS84]) and cohomology class isomorphism for the extension (Lemma 2.12). We describe these as follows.

(ND1) We will guess lists $T_1, \ldots, T_m$ of non-identity elements of $G$ where all the elements appearing in all $T_i$ are distinct (including elements in $T_i$ versus those in $T_j$, $i \neq j$), and corresponding lists $T'_1, \ldots, T'_m \subseteq H$ with $|T_i| = |T'_i|$ for all $i$ and $\sum_{i=1}^m |T_i| \in O(\log^{c+1} |G|)$.

For convenience later, we will define $T_0 = T'_0 = \emptyset$.

(For intuition, we remark that later in the proof, we will check for each $i \in [r]$, that the elements in $T_i$ induce the composition factor $G_i/G_{i-1}$ and similarly for $T'_i$ and $H_i/H_{i-1}$.)

(ND2) For each $i \in [r]$, we will guess the standard name $SN_i$ (Definition 2.6) of a finite simple group (the intended meaning of the guess here is that this corresponds to the isomorphism type of $G_i/G_{i-1}$). In addition, if the standard name includes field data for a field $\mathbb{F}_q$, we will non-deterministically guess the prime factorization of $q-1$.

(ND3) (Essentially the remaining data from Construction 2.16) For each $i \in [m]$ where $SN_i$ does not specify a Ree group ${}^2G_2(q)$:

(ND3-Gen1) Guess an SLP $S_i^{nd}$ in $\leq \sum_{j=1}^i |T_i|$ many variables (this will play the role of $S_i$ from Construction 2.16)

(ND3-Gen2) For each $\gamma \in T_i$ guess an SLP $D^{nd}(\gamma)$ in at most $|S_i^{nd}|$ many variables

(ND3-Gen3) For each $\rho \in [O(\log^c |G|)]$ guess an SLP $P^{nd}(\rho)$ in at most $|S^{nd}_{i-1}|$ many variables

(ND3-Gen4) For each $n \in T_1 \cup \cdots \cup T_{i-1}$ and each $g \in T_i$ guess an SLP $C^{nd}(n, g)$ in at most $|S^{nd}_{i-1}|$ many variables.

(ND4) For each $i \in [m]$ where $SN_i$ is a Ree group ${}^2G_2(q)$, we guess the following.

(ND4-Action) (Same as (ND3-Gen4).)

(coND4-Quotient) Our machine will co-nondeterministically try all SLPs of length $O(\log^2 n)$ in $|T_i|$ many variables, as well as all SLPs of length $O(\log^2 n)$ in $|T_1 \cup \cdots \cup T_{i-1}|$ many variables.

(coND4-CCI) The machine will co-nondeterministically try, for each pair $q, r \in {}^2G_2(q)$ and each SLP $P_{q,r}$ of length $O(\log^2 n)$ over $T_1 \cup \cdots \cup T_{i-1}$.

(coND1) (These will only be needed when we are *not* assuming all of $G$'s composition factors have uniform short presentations.) The machine will co-nondeterministically try all SLPs of length $O(\log^2 n)$ over the $T_i$. (Intention: these will be used to verify that $\langle T_1 \cup \cdots \cup T_{i+1} \rangle \neq \langle T_1 \cup \cdots \cup T_i \rangle$.)

We now show how to use these nondeterministic guesses and co-nondeterministic tries to verify whether $G \cong H$.

**Verifying standard names.** First, we verify the basic aspects of the standard names that were guessed: we are given a standard name $(L, d, q, p, f, \alpha)$, where $L$ is the label of the group, $(d, q)$ are its parameters, $q = p^e$ for some $e \geq 1$, $f$ is a polynomial in $\mathbb{F}_p[x]$ of degree $e$, and $\alpha$ is a polynomial in $\mathbb{F}_p[x]$ of degree $< e$. And we must verify the following:

(0) $L$ is a valid label, but there are only finitely many labels so this is trivial.

(1) We next check that $p$ is prime. This can be done in $\text{poly}(\log p) \leq \text{poly}(\log |G|)$ time [AKS04].[15]

(2) We next verify that $q$ is a power of $p$. The same bound from (1) in terms of $|G|$ applies to checking that $q$ is a power of $p$, because $q \leq |G|$ so $q$ is a number of at most $\log |G|$ many bits.

(3) We next check that $f(x)$ is irreducible over $\mathbb{F}_p[x]$. To verify that $f$ is irreducible we use Rabin's irreducibility test [Rab80, Lemma 1]: $f$ is irreducible if and only if $f$ divides $x^{p^e} - x$ and $gcd(f, x^{p^{e_i}} - x) = 1$ for all $e_i$ that are maximal divisors of $e$ (that is, of the form $e$ divided by a single prime). First the algorithm computes the maximal divisors $e_i$. Since $e \leq \log |G|$, the bit-length of $e$ is $O(\log \log |G|)$, so trial division lets us factor $e$ in time $\widetilde{O}(\sqrt{e}) \leq \widetilde{O}(\log^{1/2} |G|)$. The algorithm can thus find the primes $p_i$ dividing $e$, and in $O(\log^3 e) \leq O((\log \log |G|)^3)$ time can compute the maximal divisors $e_i$.

Now the algorithm checks that $f$ divides $x^{p^e} - x = x^q - x$. For this, it computes $x^{p^e} \pmod f$ by repeated squaring, taking the remainder modulo $f$ at each step to keep the degree at most $e$. If the final result is not $x$, the algorithm rejects. There are $O(\log p^e) = O(\log q) \leq O(\log |G|)$ rounds of repeated squaring, each of which takes time $\text{poly}(e, \log p) = \text{poly}(\log q) \leq \text{poly}(\log |G|)$.

[15] Alternatively, we could instead use the much older, easier, and now-classical result that PRIMES is in NP, and along with $p$ we could non-deterministically guess the witness that $p$ is prime, which would have size $\text{poly}(\log p) \leq \text{poly}(\log |G|)$. This would have the effect of relying on easier results and improving the exponent of the verification runtime, but at the expense of increasing the number of non-deterministic bits.

Finally, for each maximal divisor $e_i$ of $e$ computed above, the algorithm checks that $gcd(f, x^{p^{e_i}} - x) = 1$. Before using the standard Euclidean algorithm, as above the algorithm first computes $x^{p^{e_i}} \pmod f$ by repeated squaring, taking the remainder modulo $f$ at each step to keep the degree low. The algorithm then uses the standard Euclidean algorithm to compute $gcd(f, (x^{p^{e_i}} \pmod f) - x) = gcd(f, x^{p^{e_i}} - x)$. As these are polynomials of degree $e$, this takes at most $e$ many rounds each of which takes at most $\text{poly}(e, \log p)$ time. If the gcd is not 1, the algorithm rejects.

(4) It remains to check that $\alpha$ is a primitive root in $\mathbb{F}_p[x]/(f(x))$, that is, that $\alpha$ generates the group of units of $\mathbb{F}_p[x]/f(x) \cong \mathbb{F}_q$. For this, now that it has verified that $f$ is irreducible (and hence that $\mathbb{F}_p[x]/f(x)$ is a field of order $q = p^e$), it suffices to verify that the multiplicative order of $\alpha$ is exactly $q - 1$. As part of (ND2) we have guessed the prime factorization of $q - 1$. The algorithm first verifies this prime factorization: it checks that each listed number is prime (as above—note these numbers have at most $\log q \leq \log |G|$ many bits) and that their product is exactly $q - 1$. The algorithm then uses repeated squaring modulo $f$ (as above) to check that (1) $\alpha^{q-1} \equiv 1 \pmod f$ and (2) for each prime $p_i$ dividing $q - 1$, $\alpha^{(q-1)/p_i} \not\equiv 1 \pmod f$. This completes the verification of the basic data guessed along with the standard names.

**Verifying isomorphism.** We now proceed to checking isomorphism. First the verifier checks that the orders of the guessed simple composition factors indeed multiply to $n = |G| = |H|$. The order of a finite simple group can be computed in $\mathsf{DTISP}(\text{polylog}(n), \log(n))$ from its standard name by standard formulas (which can be found in many textbooks or, e. g., Wikipedia): $|A_k| = k!/2$, $|A_d(q)| = |PSL_{d+1}(q)| = \frac{q^{d(d+1)/2}}{gcd(d+1,q-1)} \prod_{i=1}^{d}(q^{i+1} - 1)$, etc. The algorithm then multiplies out these numbers and checks that the result is equal to $|G|$ and $|H|$, and if not, rejects. (Although this might seem like a simple check that we do "merely" for convenience, it actually turns out to be crucial.)

Now, we will proceed inductively along the composition series specified by the lists $(T_i)_{i\in[m]}$ and $(T'_i)_{i\in[m]}$; we will essentially verify that these lists indeed specify composition series for $G$ and $H$ respectively. Since the Ree groups ${}^2G_2(q)$ are not known to have a uniform short presentation, we will consider two cases: when $SN_i$ specifies a Ree group ${}^2G_2(q)$, and when $SN_i$ does not specify a Ree group.[16]

Formally, for $i \in [m]$, let $N_i = \langle T_1 \cup \cdots \cup T_{i-1} \rangle$ and $N'_i = \langle T'_1 \cup \cdots \cup T'_{i-1} \rangle$. Let $G_i = \langle N_i, T_i \rangle$ and $H_i = \langle N'_i, T'_i \rangle$. Now let $Q_i = G_i/N_i$ and $Q'_i = H_i/N'_i$. Our algorithm will verify that the map[17] $T_i \mapsto T'_i$ extends to an isomorphism between $G_i$ and $H_i$. For our base case, we define for convenience $T_0 = T'_0 = \emptyset$, and $G_0 = H_0 = \{1\}$.

For convenience, define $\Phi_i$ to be the map sending $T_j \mapsto T'_j$ for all $0 \leq j \leq i$ (see Footnote 17). If the algorithm has correctly decided that $\Phi_i$ does not extend to an isomorphism $G_i \to H_i$, then on that nondeterministic branch it rejects and need not continue further checking.

So now suppose instead that for some fixed $0 \leq i < m$, our algorithm has correctly decided that $\Phi_i$ extends to an isomorphism between $G_i$ and $H_i$. We will show that our algorithm can correctly decide whether the map $\Phi_{i+1}$ extends to an isomorphism between $G_{i+1}$ and $H_{i+1}$.

[16] And if it turns out the Uniform Short Presentation Conjecture holds for all finite simple groups, then our analysis of the case with uniform short presentations would also apply to the Ree groups.

[17] Here we think of $T_i$ and $T'_i$ as lists of elements, so when we say "the" map we mean: the one that maps the first element of $T_i$ to the first element of $T'_i$, the second element of $T_i$ to the second element of $T'_i$, and so on

- **Case 1:** Suppose that $SN_{i+1}$ does not specify a Ree group. To check that $G_{i+1}$ and $H_{i+1}$ are isomorphic, we essentially verify that the guessed data provides presentations of both $G_{i+1}$ and $H_{i+1}$ of the form given in Construction 2.16.

  The rest of the verifier's algorithm is to check that $G_{i+1}$ and $H_{i+1}$ in fact satisfy the guessed presentation. From the standard name $SN_{i+1}$, using the uniform short presentation, we construct a presentation $\langle \Gamma_{i+1} | R_{i+1} \rangle$ of the corresponding finite simple group. Precisely, we use Proposition 5.1 for $\mathrm{PSU}_3(q)$, Proposition 6.1 for $\mathrm{Sz}(q)$, and [BGK+97] for the remaining finite simple groups (excluding ${}^2G_2(q)$).

  If $|\Gamma_{i+1}| \neq |T_{i+1}|$ reject.

  Now use the SLPs guessed in (ND3) to construct the corresponding relators (Rel1)–(Rel3) from Construction 2.16; that is, use $S^{nd}_{i+1}$ in place of $S_{i+1}$ in that construction, use $D^{nd}(\gamma)$ in place of $D(\gamma)$, etc. Now we apply these relators using the guessed elements $T_{i+1}$ in place of $\widehat{\Gamma}_{i+1}$ from Construction 2.16. If any relation fails in $G_{i+1}$, reject. Similarly apply the same relators to the guessed elements $T'_{i+1}$ in place of $\widehat{\Gamma}_{i+1}$, and if any relation fails in $H_{i+1}$, reject.

  If we are not assuming all of $G$'s composition factors have uniform short presentations, then for each $1 \leq i \leq m$, let $g_i$ denote the first element of $T_i$. The machine checks the co-nondeterministic tries from (coND1) to try all words over $T_1 \cup \cdots \cup T_{i-1}$ of length $O(\log^2 |G|)$ to see if any are equal to $g_i$. If any are, then that co-nondeterministic branch rejects. (If we assume $G$'s composition factors have uniform short presentations, we do not do this step nor need these co-nondeterministic bits.)

  In total, the checks took polylogarithmic time.[18] This completes the description and complexity analysis of the algorithm in the case of having a uniform short presentation.

- **Case 2:** Suppose instead that $SN_{i+1}$ specifies ${}^2G_2(q_{i+1})$. In this case, we use the cohomological framework of Grochow and Qiao [GQ17] (recalled in Section 2.3) to decide whether $G_{i+1} \cong H_{i+1}$. Let $\alpha_i \colon G_i \to H_i$ denote the unique isomorphism extending $\Phi_i$. Let $B_{i+1}$ be the generating set for ${}^2G_2(q_{i+1})$ from Lemma 3.5, which can be computed from the standard name in $\mathsf{DTISP}(\mathrm{polylog}(n), \log(n))$. If $|T_{i+1}| \neq |B_{i+1}|$, reject.

  The algorithm proceeds with the following checks:

  - **Normality and Action Compatibility:** For each $t \in T_{i+1}$ and each $g \in T_1 \cup \cdots \cup T_i$, $tgt^{-1} \in \langle T_1 \cup \cdots \cup T_i \rangle = G_i$, we use the word $C^{nd}(g,t)$ from (ND4-Action) and verify that
    $$tgt^{-1} = C^{nd}(g,t) \qquad \Phi_{i+1}(t)\Phi_i(g)\Phi_{i+1}(t)^{-1} = C^{nd}(\Phi_i(g), \Phi_{i+1}(t)).$$
    This ensures that $G_i$ is normal in $G_{i+1}$, $H_i$ is normal in $H_{i+1}$, and that the map $\Phi_{i+1}$ (if it extends to a homomorphism) is in fact an action compatibility.
  - **Marked Isomorphism of the Quotients:** Our next step will be to verify that $G_{i+1}/G_i$ and $H_{i+1}/H_i$ are both isomorphic to ${}^2G_2(q_{i+1})$, and that the map $T_{i+1} \mapsto T'_{i+1}$ induces this isomorphism. Using the co-nondeterministically quantified SLPs from (coND4-Quotient), say we have an SLP $\sigma_1$ in $T_{i+1}$ and $\sigma_2$ in $T_1 \cup \cdots \cup T_i$. The machine checks whether $\sigma_1(T_{i+1}) = \sigma_2(T_1 \cup \cdots \cup T_i)$. If not, we consider the guesses "invalid", and that co-nondeterministic branch automatically accepts (sic! so that the

[18] And, with the possible exception of constructing the presentations according to the Uniform Short Presentation Conjecture, the rest of the procedure also took only logarithmic space.

ultimate accept/reject criterion of the earlier nondeterministic branch depends only on other co-nondeterministic branches). If $\sigma_1(T_{i+1}) = \sigma_2(T_1 \cup \cdots \cup T_i)$, then in particular, we have $\sigma_1(T_{i+1}) \in G_i$. In this case, the machine now checks deterministically that $\sigma_1(B_{i+1}) = \mathrm{Id}_7$ in $\mathrm{GL}(7, q_{i+1})$. If not, then this co-nondeterministic branch rejects (and hence, so does its parent nondeterministic branch).

– **Cohomology Class Isomorphism.** In order to check that the chosen identification $\Phi_{i+1}$ extends to a cohomology class isomorphism, we use Lemma 3.5 to get a section $s_{G,i+1} : {}^2G_2(q_{i+1}) \to G_{i+1}$ as follows. Given $M \in {}^2G_2(q_{i+1})$, we compute $\varphi^{-1}(M)$ as in Lemma 3.5 to write $M$ as a word $w_M$ in the generating set $B_{i+1}$, $w_M(B_{i+1}) = M$. Since we have identified $T_{i+1}$ with $B_{i+1}$, we then use $w_M(T_{i+1})$ as our element of $G_{i+1}$ that maps to $M$ under the given isomorphism $G_{i+1}/G_i \to {}^2G_2(q_{i+1})$. That is, we define the section $s_{G,i+1}\colon {}^2G_2(q_{i+1}) \to G_{i+1}$ by $s_{G,i+1}(M) = w_M(T_{i+1})$. On branches of the co-nondeterministic machine for $\varphi^{-1}$ that output †, our machine simply accepts, so that its ultimate accept/reject decision only relies on the co-nondeterministic branches that correctly computed $\varphi^{-1}(M)$ (and we have shown that the latter such branches indeed exist, by our definition of $\mathsf{coNTISP}(\mathrm{polylog}(n), \log n)\mathsf{MV}$ and Lemma 3.5).

To line up this final check with the statement and notation of Lemma 2.12, we introduce all the relevant notation in our present setting. Let $\alpha\colon G_i \to H_i$ be the unique isomorphism extending $\Phi_i$, and let $\beta\colon G_{i+1}/G_i \to H_{i+1}/H_i$ be the isomorphism of the composition factors induced by $\Phi_{i+1}$. Let $\theta_{G,i+1}$ be the action of $G_{i+1}/G_i$ on $G_i$ by conjugation, using the section $s_{G,i+1}$ above, and analogously for $\theta_{H,i+1}$. By Remark 2.13, we may assume the $t$ needed in the definition of pseudo-congruence (Definition 2.11) is trivial.

We then define the 2-cocycle $f_{G,i+1} : {}^2G_2(q_{i+1}) \times {}^2G_2(q_{i+1}) \to G_i$ by

$$f_{G,i+1}(q, r) = s_{G,i+1}(q) \cdot s_{G,i+1}(r) \cdot s_{G,i+1}(qr)^{-1}.$$

Define $f_{H,i+1}$ analogously. We need to verify that for all $q, r \in {}^2G_2(q_{i+1})$, the following holds:

$$\alpha(f_{G,i+1}(q, r)) = f_{H,i+1}^{1,\theta_{H,i+1}}(\beta(q), \beta(r)).$$

In order to do so, we use the co-nondeterministically tried SLPs $P_{q,r}$ from (coND4-CCI). If $P_{q,r}(T_1 \cup \cdots \cup T_{i+1}) \neq f_{G,i+1}(q, r)$, then that branch simply accepts, thus effectively leaving the final accept/reject decision in the hands of those co-nondeterministic branches in which $P_{q,r}(T_1 \cup \cdots \cup T_{i+1}) = f_{G,i+1}(q, r)$. By construction, the 2-cocycle $f_{G,i+1}(q, r)$ is in $G_i$; hence, by the Babai–Szemerédi Reachability Lemma, there is some $P_{q,r}$ satisfying the former equality. On the co-nondeterministic branch in which such a $P_{q,r}$ is found, the machine then (deterministically) checks whether $P_{q,r}(T_1' \cup \cdots \cup T_i') = f_{H,i+1}(q, r)$, and accepts if and only if they are indeed equal. Otherwise, that co-nondeterministic branch rejects.

**Correctness.** We now turn to establishing the correctness of our argument. We will show that if all of the above checks pass, then $G \cong H$; and in fact, the map $\Phi_m$ extends to an isomorphism between $G$ and $H$. For this argument, we proceed inductively along the $m$ steps of a composition series of $G$ (note that at no point did the algorithm compute such a composition series; if $G$ or $H$ does not have a composition series of length $m$, then one of the above checks will reject).

The proof is by induction on $0 \leq i \leq m$. We first consider the base case when $i = 0$. In this case, $T_0 = T'_0 = \emptyset$. So $G_0 = G'_0 = \{1\}$. Now fix $i \geq 0$, and suppose that the map $\Phi_i$ extends to an isomorphism $\alpha$ between $G_i$ and $H_i$. We will now consider the $i + 1$ case.

- **Case 1:** Suppose first that $SN_{i+1}$ is not ${}^2G_2(q)$. Because of the relations in (Rel3), we have that $G_i$ is normal in $G_{i+1} := \langle G_i, T_{i+1}\rangle$. Because of the relations in (Rel2), we have that each relator of $SN_{i+1}$ applied to $T_{i+1}$ lies in $G_i$, and hence that $G_{i+1}/G_i$ is a quotient of $SN_{i+1}$ (for convenience, we abuse notation by referring to the standard name $SN_{i+1}$ and the group it specifies interchangeably). If we are not assuming that $G$'s composition factors all have uniform short presentations, then the verification of the co-nondeterministic bits from (coND1) ensures that at least one element of $T_{i+1}$ is not in $G_i$, and hence that $G_{i+1}/G_i$ is nontrivial, and thus in fact isomorphic to $SN_{i+1}$. Similarly $\langle H_i, T'_{i+1}\rangle/H_i \cong SN_{i+1}$. The exact values of (Rel2) (not modulo $G_i$) along with the exact values of (Rel3) (and the supporting relators from (Rel1)) then imply that $G_{i+1} \cong H_{i+1}$ [BGK+97, Proof of Theorem 8.3], and this isomorphism extends the bijection $T_1 \cup \cdots \cup T_{i+1} \to T'_1 \cup \cdots \cup T_{i+1}$ described at the start of the proof of correctness. Thus, by induction, $G_{i+1} \cong H_{i+1}$, as claimed.

  Finally, by [BGK+97, Theorem 8.3], if $G_{i+1} \cong H_{i+1}$, then there are non-deterministic guesses that the algorithm will accept, namely those from Construction 2.16. Thus there exist non-deterministic guesses that are accepted by the verifier if and only if $G_{i+1} \cong H_{i+1}$.

- **Case 2:** Suppose instead that $SN_{i+1} = {}^2G_2(q_{i+1})$. The normality check above ensures that $G_i \trianglelefteq \langle G_i, T_{i+1}\rangle = G_{i+1}$ (and respectively, that $H_i \trianglelefteq \langle H_i, T'_{i+1}\rangle = H_{i+1}$).

  Next, if the checks in the Marked Isomorphism of the Quotients point pass, then by the Babai–Szemerédi Reachability Lemma, the words corresponding to the co-nondeterministically checked SLPs in which $\sigma_1(T_{i+1})$ was in $G_i$ give a presentation of $G_{i+1}/G_i$ from the generators $T_{i+1}$. The checks then imply that these relators are also valid relators for the generating set $B_{i+1}$ of ${}^2G_2(q_{i+1})$, and hence that the map defined by $T_{i+1} \mapsto B_{i+1}$ extends uniquely to a surjective homomorphism $G_{i+1}/G_i \to {}^2G_2(q_{i+1})$. Similarly, the map $T'_{i+1} \mapsto B_{i+1}$ gives a surjective homomorphism $H_{i+1}/H_i \to {}^2G_2(q_{i+1})$.

  This, on its own, is not yet enough to ensure that the preceding maps are isomorphisms, but we will now argue that if the algorithm succeeds at all stages, then in fact the preceding maps must actually be isomorphisms not merely surjections. For, if at all stages $i$ the checks pass, then for the non-Ree-group composition factors $G_{i+1}/G_i$ the machine will have verified that they were indeed isomorphic to the guessed names $SN_{i+1}$. For the factors such that $SN_{i+1}$ was guessed to be a Ree group, what the checks have verified is that $G_{i+1}/G_i$ surjects onto the guessed ${}^2G_2(q_{i+1})$. In particular, this implies that $|G_{i+1}/G_i| \geq |{}^2G_2(q_{i+1})|$. But since this is true for all $i$, and the machine has also verified (at the very start of the verification) that $\prod_i |SN_i| = |G|$, we cannot have $|G_{i+1}/G_i| > |SN_{i+1}| = |{}^2G_2(q_{i+1})|$, for then we would have $|G| = \prod_i |G_i/G_{i-1}| > \prod_i |SN_i| = |G|$, a contradiction. Thus, if all checks at all stages passed, then it must have in fact been the case that the maps $T_{i+1} \mapsto B_{i+1}$ extended uniquely not just to surjective homomorphisms, but in fact to *isomorphisms* $G_{i+1}/G_i \xrightarrow{\cong} {}^2G_2(q_{i+1})$, as desired.

  Since this was also true for the maps $T'_{i+1} \mapsto B_{i+1}$, composing one of these isomorphisms with the inverse of the other also ensure that the map $T_{i+1} \mapsto T'_{i+1}$ extends uniquely to an isomorphism of composition factors $G_{i+1}/G_i \to H_{i+1}/H_i$.

We now have isomorphisms $\alpha : G_i \cong H_i$, $\beta : G_{i+1}/G_i \cong H_{i+1}/H_i$, and sections $s_{G,i+1} : G_{i+1}/G_i \to G_{i+1}$ and $s_{H,i+1} : H_{i+1}/H_i \to H_{i+1}$ induced by $T_{i+1}, T'_{i+1}$ respectively. The correctness for the remainder of this case follows from Lemma 2.12.

Thus, by induction, our algorithm correctly determines whether $G \cong H$.

**Complexity.** At each stage, we use polylog$(n)$ existentially-quantified non-deterministic bits. At a stage when we are when contending with ${}^2G_2(q)$ (Case 2), we also require polylog$(n)$ universally-quantified co-nondeterministic bits. Furthermore, the deterministic computation is $\mathsf{DTIME}(\text{polylog}(n))$-computable. As $m \leq \lceil \log |G| \rceil$, we obtain our bound of

$$\exists^{\text{polylog}(n)} \forall^{\text{polylog}(n)} \mathsf{DTIME}(\text{polylog}(n)),$$

as desired.

Lastly, we note that if there were a uniform short presentation for ${}^2G_2(q)$—or more generally if all of $G$'s composition factors have uniform short presentations—then we could handle each composition factor as described in Case 1 *and we can omit the use of the co-nondeterministic bits (coND1).* For in that case, we have verified that each $G_{i+1}/G_i$ is a quotient of the simple group $SN_{i+1}$. And furthermore, since at the start we verified that $\prod_i |SN_i| = |G|$, if any of the quotients $G_{i+1}/G_i$ were trivial, we would get $|G| = \prod |SN_i| > \prod_i |G_{i+1}/G_i| = |G|$, a contradiction. Thus, the guesses above indeed verified the isomorphisms $G_{i+1}/G_i \cong SN_{i+1} \cong H_{i+1}/H_i$, as claimed.[19] Thus, if all of $G$'s composition factors are assumed (or promised) to have uniform short presentation then isomorphism between two groups can be decided in

$$\exists^{\text{polylog}(n)} \mathsf{DTIME}(\text{polylog}(n)),$$

as desired. □

**Remark 3.6** (On uniformity)**.** As is visible in the structure of the above proof, if we assume only the (non-uniform) Short Presentation Conjecture, then the same proof gives a non-uniform analogue of Theorem 3.1, namely that Group Isomorphism is solvable by non-uniform circuits that are conjunctions of a quasipolynomial-size DNF and a polynomial-size CNF. Furthermore, if the uniformity in the Uniform Short Presentation Conjecture is upgraded from $\mathsf{DTIME}(\text{polylog}(n))$ to $\mathsf{DTISP}(\text{polylog}(n), \log(n))$, then Proposition 3.4 upgrades by replacing $\mathsf{DTIME}(\text{polylog}(n))$ with $\mathsf{DTISP}(\text{polylog}(n), \log(n))$. In the non-uniform setting, the non-uniform advice at length $n$ consists of short presentations of all finite simple groups of order $\leq n$; note that this only depends on $n$ and not on the input groups $G, H$. Since there are at most two isomorphism types of simple groups of each order (and none of odd order, by the Feit–Thompson Odd Order Theorem), there are at most $n$ such groups (and in fact many fewer, but this bound suffices for the argument), and the total number of bits for this advice is $O(n \,\text{polylog}(n))$. When the $\mathsf{DTIME}(\text{polylog}(n))$ machine would need to convert a standard name $SN_i$ into a short presentation, it instead queries that short presentation from the advice.[20]

[19] The reason we need the co-nondeterministic bits (coND1) and their verification when the groups can have both Ree and non-Ree composition factors—and without using uniform short presentations for the Ree factors—is that our other verifications ensure that in the non-Ree case $|G_{i+1}/G_i|$ is no bigger than $|SN_{i+1}|$, while in the Ree case $|G_{i+1}/G_i|$ is no smaller than $|SN_{i+1}|$, and it is in principle possible that the checks could pass incorrectly with these two balancing out. But if we assume uniform short presentations for the Ree factors as well, or if the group does not have Ree factors, then our verifications even without (coND1) would ensure that $|G_{i+1}/G_i| \leq |SN_{i+1}|$ for *all* $i$, and hence that each successive quotient must in fact be isomorphic to the guessed simple group.

[20] For these purposes, we may even assume that the $\mathsf{DTIME}(\text{polylog}(n))$ machine has "associative memory" access

# 4 Lower Bounds

Under the Uniform Short Presentation Conjecture, the upper bound of Section 3 expresses Group Isomorphism as the conjunction of a polynomial-size CNF checking validity of the input Cayley tables and a quasipolynomial-size DNF deciding isomorphism under that promise. This suggests asking whether either component can be replaced by a formula of the other type, thus yielding circuits of depth 2. In Section 4.1, we show that the validity CNF cannot be replaced by a quasipolynomial-size DNF: any DNF deciding validity, and hence any DNF deciding total Group Isomorphism, requires at least $2^{n \log n - O(n)}$ terms for groups of order $n$. For the other "half," we prove in Section 4.2 an $n^{\Omega(\log n)}$ lower bound for CNFs deciding the promise version of Group Isomorphism. This rules out polynomial-size CNFs but does not rule out quasipolynomial-size CNFs. Section 4.3 discusses the significance of such an upper bound and explains an obstruction to obtaining a stronger lower bound.

## 4.1 Exponential DNF lower bound

We get our exponential lower bound on DNFs for Group Isomorphism from the same lower bound on DNFs for Validity Testing: testing whether a given multiplication table is in fact the multiplication table of a group. Although one might feel that a lower bound on checking validity of multiplication tables hardly gets at the "heart" of the Group Isomorphism problem, we start with an observation that, for DNF complexity, the Uniform Short Presentation Conjecture implies that the DNF complexity of GpI and of Validity Testing are polynomially equivalent:

**Observation 4.1.** *1. Validity testing of Cayley tables reduces to* GpI *by a* DLOGTIME*-uniform projection reduction. Testing whether a multiplication table is a valid quasigroup (=Latin square) similarly reduces to* Quasigroup Isomorphism *and* Latin Square Isotopy*.*

*2. Assuming the Uniform Short Presentation Conjecture, if* Validity Testing *is decidable by (uniform) DNFs of size $f(n)$, then* Group Isomorphism *is decidable by (uniform) DNFs of size $f(n)^2 n^{\mathrm{polylog}(n)}$.*

We use Observation 4.1(1) for our unconditional lower bound below; part (2) is simply to show that for DNF complexity above quasipolynomial (which this is: we will show it is in fact exponential), this approach is essentially—up to cubing the size, and the Uniform Short Presentation Conjecture—without loss of generality.

*Proof.* 1. If $T$ is a multiplication table, then $(T, T)$ is a yes-instance of GpI if and only if $T$ is a valid multiplication table of a group. (The same proof works for quasigroups, *mutatis mutandis*.) Thus if GpI has DNFs of size $O(f(n))$, then so does Validity Testing.

2. In the opposite direction, assuming the Uniform Short Presentation Conjecture, by Fact 2.3 and the second half of Theorem 3.1, there is a uniform DNF $D_n$ of size $n^{\log^c n}$ such that, whenever $T_1$ and $T_2$ are valid Cayley tables of groups of order $n$, $D_n(T_1, T_2) = 1$ if and only if $T_1 \cong T_2$. Consequently, on arbitrary pairs of multiplication tables,

$$\textsc{Group Isomorphism}_n(T_1, T_2) = \mathrm{VALID}_n(T_1) \wedge \mathrm{VALID}_n(T_2) \wedge D_n(T_1, T_2).$$

to the advice, and can literally look up the presentation by name, rather than worrying about where it is placed in terms of bits. For when it is then converted into a non-uniform Boolean formula in terms of DNFs and CNFs, those wires will simply have hard-coded 0s and 1s coming from the short presentations.

Thus if Validity Testing has DNFs of size $f(n)$, then we can distribute the triple-$\wedge$ over the DNFs to get a single DNF of size $f(n)^2 n^{\log^c n}$. □

**Theorem 4.2.** *Any (uniform or non-uniform) DNF on multiplication tables of order $n$ that is 1 on all Cayley tables of groups and 0 on all other multiplication tables must have at least $n!/n^{\log n} \geq 2^{n \log n - O(n)}$ terms. Hence the same lower bound holds on DNFs for* GpI.

*For Latin squares instead of groups, we get an analogous lower bound of $(n!)^{2n}/n^{n^2} \geq 2^{n^2 \log n - O(n^2)}$.*

Note that the number of input variables in either case is $n^2\lceil \log_2 n \rceil =: N$; as a function of the number of input variables the lower bound for groups is $2^{\Omega\left(\sqrt{N \log N}\right)}$, while for quasigroups it is $2^{N(1-1/\log N)}$, nearly maximal.

*Proof.* The key fact is that there are no two multiplication tables of quasigroups that differ in only one entry, hence *a fortiori* they also cannot differ in only one bit. For if $T$ is the multiplication table of a (quasi)group and $T'$ differs from $T$ in only one entry, say $T(x, y) \neq T'(x, y)$, then the $x$-th row of $T'$ is not a permutation. (If we remove one entry from a row, the value of that entry is uniquely determined as the only entry missing from the rest of the row.)

Now suppose $\varphi$ is a DNF that decides validity. The total number of variables is $N = n^2\lceil \log_2 n \rceil$. If any term $t$ of the DNF has width $< N$, then $t$ is 1 on a sub-cube of dimension $\geq 1$, contradicting the fact from the previous paragraph.

Thus every term has width exactly $N$, hence is 1 on exactly one input, so the number of terms must be equal to the number of valid tables.

In the case of groups, each group of order $n$ has automorphism group of size at most $n^{\log n}$, so there are at least $n!/n^{\log n}$ distinct tables corresponding to each group. Thus the total number of valid tables is at least $n!/n^{\log n} \geq 2^{n \log n - O(n)}$.[21]

In the case of quasigroups, the number of Latin squares of order $n$ is at least $(n!)^{2n}/n^{n^2}$ (e. g., [vLW01, Thm. 17.2]), so we get the same lower bound on the number of terms of a DNF that decides Latin squares. □

## 4.2 Quasi-polynomial CNF lower bound

In this section we prove:

**Theorem 4.3.** *Any (uniform or non-uniform) CNF solving the promise* Group Isomorphism *problem on multiplication tables of groups of order $n = 2^{2k}$ has size at least $n^{\Omega(\log n)}$.*

The groups we use for the lower bound will be Abelian of exponent 4, and the constant in the exponent is at least 1/16.

*Proof idea.* The proof builds groups $G_M$ from $k \times k$ binary matrices $M$, so that the isomorphism type of $G_M$ is completely determined by $\mathrm{rank}(M)$. Thus Matrix Rank reduces (with exponential size increase) to Group Isomorphism; in particular, Group Isomorphism contains within it the problem of distinguishing matrices of rank $k/2$ from matrices of rank $< k/2$. When a CNF clause is restricted to these Cayley tables, its falsifying set is defined by affine-linear constraints on the rows of $M$, with distinct rows having independent constraints; such a set is called a row-cube. If such a row-cube avoids all rank $k/2$ matrices but covers some rank $k/2 - 1$ matrix, then it cannot be too

[21] When $n = p^k$ and $p = O(1)$, the number of isomorphism types is $n^{\Omega(\log^2 n)}$ [Hig60, Sim65], but this only affects the lower-order terms in the exponent of the asymptotic count of the number of valid Cayley tables.

large. But there are exponentially many rank $k/2-1$ matrices, so covering all of them requires $2^{\Omega(k^2)}$ clauses. Since $k = \Theta(\log n)$, this is exactly $n^{\Omega(\log n)}$. □

Throughout the following, $k$ will denote a natural number and $M$ will denote a $k \times k$ matrix. We build a group $G_M$ from such a matrix $M$ as follows.

**Construction 4.4.** Given $M \in \mathrm{Mat}_k(\mathbb{F}_2)$, the elements of $G_M$ are pairs $(u, z) \in \mathbb{F}_2^k \times \mathbb{F}_2^k$, so $|G_M| = 2^{2k}$. Multiplication is defined by $(u, z) \cdot_M (v, w) = \big(u+v,\ z+w+M(u \odot v)\big)$, where $u \odot v$ denotes the coordinate-wise (Hadamard) product, $(u \odot v)_i = u_i v_i$.

We use Matrix Rank over a field $\mathbb{F}$ to refer to the language

$$\{(M, r) : M \text{ is a square matrix over } \mathbb{F} \text{ and } \operatorname{rank}(M) = r\}.$$

We use $I_r \oplus 0_{k-r}$ to denote the $k \times k$ diagonal matrix $\mathrm{diag}(1, 1, \ldots, 1, 0, 0, \ldots, 0)$ of rank exactly $r$.

**Proposition 4.5.** *The group $G_M$ of Construction 4.4 is an Abelian 2-group of exponent at most 4, with isomorphism type $G_M \cong (\mathbb{Z}_4)^{\operatorname{rank}(M)} \times (\mathbb{Z}_2)^{2k-2\operatorname{rank}(M)}$. In particular, $G_M \cong G_N \iff \operatorname{rank}(M) = \operatorname{rank}(N)$.*

*Furthermore, the function $(M, r) \mapsto (G_M, G_{I_r \oplus 0_{k-r}})$ is an $\mathbb{F}_2$-affine projection reduction from* Matrix Rank *over $\mathbb{F}_2$ to* Group Isomorphism*, in which each bit of the output depends on at most one row of $M$.*

**Remark 4.6** (On $\mathsf{AC}^0$ reductions from Parity)**.** Since depth-$d$ $\mathsf{AC}$ circuits for $n$-bit Parity requires size $2^{\Omega(n^{1/(d-1)})}$ [Hås86], it follows from [CTW13] unconditionally that any $\mathsf{AC}^0$ reduction from Parity to Group Isomorphism on groups of order $n$ can be from parities on at most $\mathrm{polylog}(n)$ bits; the tighter upper bound of depth-4 circuits of size $n^{O(\log n)}$ from [CGLW25] implies such a reduction can be from parities on at most $O(\log^6 n)$ bits. However, we do not know a reduction from Parity on more than $O(\log n)$ bits to GpI for groups of order $n$, and such a reduction gives no notable lower bounds even on CNFs, since all functions on $O(\log n)$ bits have (at least non-uniform; and in the case of Parity these can be made uniform) decision trees of depth $O(\log n)$, hence CNFs and DNFs of $\mathrm{poly}(n)$ size. In contrast, we give a reduction from $k \times k$ Matrix Rank over $\mathbb{F}_2$ to GpI on groups of order $n = 4^k$. Whereas Parity is in logspace, Matrix Rank over $\mathbb{F}_2$ is $\mathsf{Mod}_2\mathsf{L}$-complete [BDHM92].[22] In particular, $k$-bit Parity reduces to $k \times k$ Matrix Rank over $\mathbb{F}_2$, so our reduction still does not reduce from a parity on more than $O(\log n)$ bits. By reducing from the harder problem of Matrix Rank instead, we still incur an exponential size increase, but are able to get our quasi-polynomial lower bound.

*Proof of Proposition 4.5.* A direct computation shows $(u, z)^2 = (0, Mu)$, so the subgroup of squares is $G_M^{(2)} := \{x^2 : x \in G_M\} = \{(0, Mu) : u \in \mathbb{F}_2^k\} = \{0\} \times \mathrm{im}(M)$, and every element has order dividing 4. If $\operatorname{rank}(M) = r$, then $\mathrm{im}(M)$ is an $r$-dimensional subspace over $\mathbb{F}_2$, giving $|G_M^{(2)}| = 2^r$. The square subgroup detects the number of $\mathbb{Z}_4$-factors: in $G_M$ that number is exactly $\operatorname{rank}(M)$. The stated isomorphism type follows, and two such groups are isomorphic if and only if they have the same rank.

[22] Although they phrase their results in terms of $\mathsf{NC}^1$ reductions, the reductions are in fact already in $\mathsf{AC}^0$ (in fact, we believe even in $\mathsf{NC}^0$, but we do not need that stronger statement for our discussion here so we have not verified it carefully). The relevant reductions are the $\mathsf{Mod}_2\mathsf{L}$-completeness of what they call the Mod-2-Gap problem, and the reduction from the latter to Matrix Rank, which follows immediately from their Proposition 9.

We have already established that the function $(M, r) \mapsto (G_M, G_{I_r \oplus 0_{k-r}})$ is a many-one reduction (of exponential stretch) from Matrix Rank over $\mathbb{F}_2$ to Group Isomorphism. It remains to show that this reduction is an $\mathbb{F}_2$-affine projection reduction in which each bit of the output depends on at most one row of $M$. For this, we examine the definition of the product in $G_M$: $(u, z) \cdot_M (v, w) = (u + v, z + w + M(u \odot v))$. The first component, $u + v$, does not depend on $M$, so its coordinates are in fact constants. The $j$-th coordinate of the second component is

$$z_j + w_j + \sum_{i=1}^{k} M_{ji}\, u_i v_i,$$

which depends only on row $j$ of $M$, and for each fixed pair $(u, v)$, is $\mathbb{F}_2$-affine linear in that row of $M$. □

**Remark 4.7** (Odd characteristic)**.** This particular construction only works in characteristic 2. The same construction in characteristic $p > 2$ has $(u, z)^i = \left(iu, iz + \binom{i}{2} M(u \odot u)\right)$ by induction on $i$, and hence $(u, z)^p = (0, 0)$ if $p > 2$, since $\binom{p}{2}$ is divisible by $p$ when $p > 2$, so the resulting group is always an elementary Abelian group of exponent $p$. For odd $p$ we can nonetheless get a similar result using Baer's correspondence [Bae38]: given a $k \times k$ matrix $M$ over $\mathbb{F}_p$, we may define the group $G_{M,p}$ whose underlying set is $\mathbb{F}_p^{2k} \times \mathbb{F}_p$ with multiplication defined by $(u, \alpha) \cdot (v, \beta) = \left(u + v, \alpha + \beta + \frac{1}{2} u^\top \begin{pmatrix} 0 & M \\ -M^\top & 0 \end{pmatrix} v\right)$. This group is isomorphic to the direct product of the Heisenberg group of order $p^{2\,\mathrm{rank}(M)+1}$ (which is non-Abelian and directly indecomposable) and the Abelian group $\mathbb{Z}_p^{2(k-\mathrm{rank}(M))}$. However, this is no longer an affine projection because it requires encoding mod-$p$ arithmetic using Boolean gadgets. To get around the latter issue, one could instead consider "CNFs over mod-$p$ atoms", in which we allow atomic formulae $[x_i = \alpha]$ where $\alpha \in \mathbb{F}_p$ to take the place of literals, and then build CNFs on top of those atoms. In that case, this again becomes an $\mathbb{F}_p$-affine projection reduction, and the subsequent arguments will then show that "CNFs over mod-$p$ atoms" that solve promise Group Isomorphism must have size at least $n^{\Omega(\log n)}$.

### 4.2.1 Additional Preliminaries: Row-Cubes

For our lower bound, we view the matrix space as a product over its rows. Namely, we identify $\mathbb{F}_2^{k \times k} \cong \left(\mathbb{F}_2^k\right)^k$ by writing a matrix $M$ as its ordered tuple of rows $M = (m_1, \ldots, m_k)$ where $m_j \in \mathbb{F}_2^k$. We call a set of the form: $F = A_1 \times \cdots \times A_k$ where each $A_j \subseteq \mathbb{F}_2^k$ is an affine subspace, a row-cube. Such products are cubes in the terminology of [BDPW10] with the additional property of affineness. They are also special cases of cylinder intersections which are mostly used in multi-party communication complexity: indeed, $A_1 \times \cdots \times A_k = \bigcap_{i=1}^{k} \left(A_1 \times \cdots \times A_{i-1} \times \mathbb{F}_2^k \times A_{i+1} \times \cdots \times A_k\right)$ where the $i$th set in the intersection is a cylinder in the $i$th row-coordinate. We use the name row-cubes to emphasize the row-wise product structure.

Row-cubes have three elementary properties that we will use repeatedly. First, they are themselves affine subspaces of $\mathbb{F}_2^{k \times k}$. Indeed, if $A_j = a_j + U_j$, where $U_j$ is a linear subspace of $\mathbb{F}_2^k$, then $F = A + U$, $U = U_1 \times \cdots \times U_k$, where $A$ is the matrix whose $j$-th row is $a_j$. Consequently, $\dim F = \sum_{j=1}^{k} \dim U_j$ and $|F| = 2^{\dim F}$.

Second, any two matrices in a row-cube can be joined by changing one row at a time. More precisely, let $X = (x_1, \ldots, x_k)$, $Y = (y_1, \ldots, y_k)$ belong to $F$. For $t = 0, \ldots, k$, define $X^{(t)} := (y_1, \ldots, y_t, x_{t+1}, \ldots, x_k)$. Every $X^{(t)}$ belongs to $F$, since its $j$-th row belongs to $A_j$: it is either $y_j \in A_j$ or $x_j \in A_j$. Thus $X = X^{(0)}, X^{(1)}, \ldots, X^{(k)} = Y$ is a path inside $F$ in which consecutive matrices differ in at most one row.

We will also use the following theorem

**Theorem 4.8** ([dSP10, Theorem 6], cf. [Mes85])**.** *Over an arbitrary field $\mathbb{F}$, if $\mathcal{A}$ is an affine subspace of $\mathbb{F}^{k\times k}$ and every matrix in $\mathcal{A}$ has rank at most $s$, then* $\dim \mathcal{A} \leq ks$.

### 4.2.2 Returning to the lower bound

**Lemma 4.9.** *Fix $r \leq k \in \mathbb{N}$. After restricting to the family of pairs of multiplication tables $\{(G_M, G_{I_r \oplus 0_{k-r}}) : M \in Mat_k(\mathbb{F}_2)\}$, the falsifying set of each CNF clause has the form $F = A_1 \times A_2 \times \cdots \times A_k$, where each $A_j \subseteq \mathbb{F}_2^k$ is an affine subspace of possible choices for row $j$ of $M$. Thus $F$ is a row-cube in the sense defined above.*

*Proof.* By the second half of Proposition 4.5, each bit of the multiplication table of $G_M$ (in our chosen way of representing elements) depends $\mathbb{F}_2$-affine-linearly on at most one row of $M$. Therefore each input bit read by a clause is either a constant or an affine linear function of exactly one row of $M$, and the falsifying set of the clause imposes independent affine linear constraints on each row. □

**Lemma 4.10.** *Let $F = A_1 \times \cdots \times A_k$ be a row-cube over a field $\mathbb{F}$. Then the set of ranks of matrices occurring in $F$ is a contiguous set of integers.*

*In particular, if $F$ contains at least one rank-$s$ matrix but no rank-$(s+1)$ matrix, then every matrix in $F$ has rank at most $s$. Consequently, such $F$ satisfy* $\dim_{\mathbb{F}} F \leq ks$.

*Proof.* Suppose $X, Y \in F$ have ranks $r = \operatorname{rank}(X) < \operatorname{rank}(Y) = r'$. By the second property of the row-cubes as described above, there is a sequence $X = X^0, X^1, \ldots, X^k = Y$ of matrices in $F$ such that consecutive matrices differ in at most one row. Consequently their ranks differ by at most one. Thus every rank in between $r$ and $r'$ must occur as the rank of at least one matrix in the sequence $X^0, \ldots, X^k \in F$. This establishes the first claim.

The "in particular" then follows immediately. Once we have $\max_{X\in F} \operatorname{rank}(X) \leq s$, [dSP10, Theorem 6] implies that $\dim F \leq ks$, completing the proof of the lemma. □

The following estimate is standard, but crucial for our lower bound:

**Lemma 4.11.** *Over $\mathbb{F}_q$, there are at least $q^{r(2k-r)}$ matrices of size $k \times k$ and rank at most $r$.*

*Proof.* The matrix product function $\mathbb{F}_q^{k\times r} \times \mathbb{F}_q^{r\times k} \to \mathbb{F}_q^{k\times k}$ has image consisting exactly of the matrices of rank $\leq r$. Moreover, the pre-image of any such matrix is a set of the form $\{(Ag, g^{-1}B) : g \in \mathrm{GL}(r, q)\}$, so the number of $k \times k$ matrices of rank at most $r$ is exactly $q^{2kr}/|\mathrm{GL}(r, q)|$. Since $|\mathrm{GL}(r, q)| \leq q^{r^2}$, the result follows.[23] □

Finally, we now have all the ingredients for the proof of our main CNF lower bound, Theorem 4.3:

[23] Although this upper bound is not quite tight, there is also a lower bound of $q^{r^2-r}$, so it is not very far from tight. Thus, even using a tighter upper bound could at most affect the lower-order terms in the exponent of our final lower bound.

*Proof of Theorem 4.3.* Let $C = \Gamma_1 \wedge \cdots \wedge \Gamma_m$ be a CNF solving the promise GPI problem. For a matrix $M$ and integer $r = s+1$, we consider applying $C$ to the pair $(G_M, G_{I_r \oplus 0_{k-r}})$ (the output of the reduction from Proposition 4.5). Let $F_i = \{M : \Gamma_i(G_M, G_{I_r \oplus I_{k-r}}) = 0\}$ be the set of matrices $M$ such that the output of the reduction from Proposition 4.5 falsifies the clause $\Gamma_i$. By Lemma 4.9 each $F_i$ is a row-cube. Correctness of $C$ implies:

1. No clause is false on a rank-$r$ matrix: $F_i \cap \{M : \mathrm{rank}(M) = r\} = \varnothing$ for every $i$.
2. Every matrix of rank $\neq r$ falsifies some clause, in particular, $\{M : \mathrm{rank}(M) \leq r-1 = s\} \subseteq \bigcup_{i=1}^{m} F_i$.[24]

Condition (1) says that each $F_i$ contains no rank-$r$ matrix. Thus each $F_i$ that covers at least one rank-$s$ matrix satisfies the hypothesis of Lemma 4.10, giving $|F_i| \leq 2^{ks}$. The $F_i$ that contain only smaller-rank matrices are even smaller (at most $2^{k(s-1)}$), and those that contain no matrices of rank $< r$ do not contribute to the sum at all. Next, condition (2) gives us that

$$|\{M : \mathrm{rank}(M) \leq s\}| \leq \sum_{i=1}^{m} F_i \leq \sum 2^{ks} = m2^{ks}.$$

By Lemma 4.11, the left-hand side here is at least $2^{s(2k-s)}$. Putting these together we get

$$2^{s(2k-s)} \leq m2^{ks}$$

hence $m \geq 2^{ks-s^2}$. Since our CNF must work for all ranks $s$, we may choose $s$ to optimize this quantity, which happens at $s = k/2$. Choosing $s = k/2$, we get $m \geq 2^{k^2/2-(k^2/4)} = 2^{(1/4)k^2}$. Since $n = 2^{2k}$, we have $k = (1/2)\log_2 n$, hence $m \geq n^{\frac{1}{16}(\log_2 n)} \geq n^{\Omega(\log n)}$. □

**Remark 4.12.** There is a matching $n^{O(\log n)}$ upper bound only in the canonical coordinate presentation, where the elements are explicitly labeled as pairs $(u, z) \in \mathbb{F}_2^k \times \mathbb{F}_2^k$. In that presentation, one can read off $M_{ji}$ from $(e_i, 0)^2 = (0, Me_i)$. But this is not yet an arbitrary Cayley table upper bound, because under an arbitrary relabeling the formula does not know which table element is $(e_i, 0)$, nor how to decode the coordinates of the element corresponding to $(0, Me_i)$.

Construction 4.4 has a natural tensor analogue; although one might hope to use a tensor analogue to improve the lower bound, in the remainder of this section we discuss a significant obstacle to this approach.

**Definition 4.13.** Let $p$ be an odd prime. For class-2 exponent-$p$ Baer groups, the parameter space is the space of skew-symmetric matrix tuples $X = \left(\Lambda^2 V^*\right)^\ell$, $V = \mathbb{F}_p^k$. An element $T \in X$ is an $\ell$-tuple of skew-symmetric matrices $T = (M^1, \ldots, M^\ell)$, or equivalently a skew-symmetric bilinear map $b_T : V \times V \to \mathbb{F}_p^\ell$. The associated Baer group is $G_T = V \times \mathbb{F}_p^\ell$ with multiplication $(v, z)(v', z') = \left(v + v',\ z + z' + \frac{1}{2} b_T(v, v')\right)$.

**Proposition 4.14** (Baer's Correspondence [Bae38])**.** *Let $p$ be an odd prime. Let $H = \mathrm{GL}_\ell(p) \times \mathrm{GL}_k(p)$ act on $X$ by simultaneous change of basis in $\mathbb{F}_p^\ell$ and in $V$. Then $G_T \cong G_S \iff T \cong S \iff S = h \cdot T$ for some $h \in H$.*

[24] One might be able to sharpen the argument a bit by including here also the matrices of rank $> r$, but that can at most improve the constant in the exponent: for even if we replace our lower bound on $|\bigcup F_i|$ with the the size of the set of all $k \times k$ matrices, our final lower bound would still be of the form $2^{\Omega(k^2)}$, which is asymptotically what we already achieve.

Proposition 4.14 plays the role that the rank classification plays in the matrix setting: it reduces Group Isomorphism to a natural orbit problem. The Skew-Symmetric (or Alternating) Matrix Space Isometry problem is accordingly to decide whether a given pair $(T, S) \in X \times X$ satisfies $T \cong S$; that is, the yes-set is $\mathcal{Y} = \{(T, S) \in X \times X : T \cong S\} = \{(T, h \cdot T) : T \in X,\ h \in H\}$.

**Remark 4.15** (Obstacle to using tensors in our lower bound strategy)**.** There is an important structural difference between the matrix and tensor settings. A fixed tensor orbit has size at most $|\operatorname{Orb}(R)| \le |\mathrm{GL}_\ell(p)|\,|\mathrm{GL}_k(p)| = p^{O(k^2+\ell^2)}$. In the balanced regime $\ell \asymp k$, this is only $p^{O(k^2)}$, whereas the full tensor space has size $|X| = p^{\ell\binom{k}{2}} = p^{\Theta(k^3)}$. Thus, a fixed tensor orbit is asymptotically thinner (in the exponent) than a generic rank layer in the matrix setting (viz. there are $p^{k^2}$ matrices of size $k \times k$ and $p^{\Theta(k^2)}$ such matrices of rank $k/2$). For this reason, the natural object for a covering argument is not a single fixed orbit in $X$ (in the rank setting we considered only the orbit of a rank-$k/2$ matrix), but rather the two-input isomorphism relation $\mathcal{Y} := \{(T, S) \in X \times X : T \cong S\}$.

One might then try to imitate the rank argument directly on $X \times X$. In canonical Baer labels, Cayley-table entries are affine-linear functions of the tensor coordinates, so the falsifying set of a width-$w$ CNF clause is, as in the matrix case, an affine subspace of $X \times X$ of codimension at most $w$. A tensor analogue of the rank lower bound would therefore follow from showing that every sufficiently large affine subspace of $X \times X$ intersects $\mathcal{Y}$; equivalently, that every yes-free affine subspace has large codimension or covers very little mass. This strategy, however, fails on the full space $X \times X$. Indeed, assume that $k$ is even, fix a nonzero vector $v \in V$, and fix a nondegenerate skew-symmetric form $J \in \Lambda^2 V^*$. Let $\mathcal{A} \subseteq X$ be the linear subspace consisting of tensors $T$ for which $v \in \operatorname{rad}(T)$. This imposes $\ell(k-1)$ linear constraints. Let $\mathcal{B} \subseteq X$ be the affine subspace consisting of tensors $S = (N_1, \dots, N_\ell)$ with $N_1 = \sum_{i=1}^{k/2}(E_{2i,2i+1} - E_{2i+1,2i})$, the standard full-rank skew-symmetric matrix. This has codimension $\binom{k}{2}$, and every $S \in \mathcal{B}$ has trivial radical. Consequently, $\mathcal{A} \times \mathcal{B}$ is a yes-free affine subspace of $X \times X$ (because the radical is an isomorphism invariant) of codimension

$$\ell(k-1) + \binom{k}{2} = O(k^2)$$

in the balanced regime $\ell \asymp k$. Thus the tensor analogue of the rank argument cannot be applied directly to all of $X \times X$.

## 4.3 QuasiCNF implies Group Isomorphism is in coNP

**Observation 4.16.** $\forall^{\log^c n}\mathsf{DTIME}(\mathrm{polylog}(n)) \subseteq \mathsf{coNP}$.

*Proof.* $\forall^{\log^c n}\mathsf{DTIME}(\mathrm{polylog}(n)) \subseteq \forall^{\mathrm{poly}(n)}\mathsf{DTIME}(\mathrm{polylog}(n)) \subseteq \forall^{\mathrm{poly}(n)}\mathsf{P} = \mathsf{coNP}$. □

As a consequence, if we were able to solve Group Isomorphism in

$$\forall^{\mathrm{poly}(\log n)}\mathsf{DTIME}(\mathrm{polylog}(n)),$$

then combining with Lemma 3.3 would place GpI into coNP, which remains a significant open question. Arvind and Torán [AT11] showed that in the case of solvable groups, GpI is *almost* in $\mathsf{NP} \cap \mathsf{coNP}$. Precisely, they exhibited an Arthur–Merlin protocol for Group Non-Isomorphism where Arthur used $O(\log^6 n)$ bits and Merlin used $O(\log^2 n)$ bits. Additionally, Arvind and Torán showed that if $\mathsf{EXP} \not\subseteq \mathsf{i.o.\text{-}PSPACE}$, then GpI for solvable groups belongs to $\mathsf{NP} \cap \mathsf{coNP}$. Arvind and Torán also note that the Uniform Short Presentation Conjecture remains a key obstacle for extending their work to all finite groups.

# 5 Uniformity of the Hulpke–Seress Short Presentations for ${}^2A_2(q) = \mathrm{PSU}_3(q)$

**Proposition 5.1.** *Let $q = p^e > 2$, and let $G := {}^2A_2(q) = \mathrm{PSU}_3(q)$. Given the standard name of $G$, one can construct, in $\mathsf{DTIME}(\mathrm{polylog}(|G|))$, a presentation of $G$ of word length $O(\log^2 q) = O(\log^2 |G|)$.*

## 5.1 Additional preliminaries for $\mathrm{PSU}_3(q)$

In this section, we will recall additional preliminaries concerning $\mathrm{PSU}_3(q)$. We refer to [HS01] for a reference.

Let $p$ be a prime, and let $q = p^e$ for some $e > 0$. The group $\mathrm{SU}_3(q)$ is a subgroup of $\mathrm{SL}(3, q^2)$, and $\mathrm{PSU}_3(q) = \mathrm{SU}_3(q)/Z(\mathrm{SU}_3(q))$. The center of $\mathrm{SU}_3(q)$ consists of scalar matrices, hence it is either trivial or has order 3 (iff $3|q+1$). For $a \in \mathbb{F}_{q^2}$, we define the trace and norm maps by:

$$\mathrm{Tr}(a) := a + a^q, \quad \mathrm{Nm}(a) := \mathrm{Nm}_{\mathbb{F}_{q^2}/\mathbb{F}_q}(a) = a^{q+1}.$$

Note that, as $\mathbb{F}_{q^2}$ is a degree-2 extension of $\mathbb{F}_q$, the automorphism $a \mapsto a^q$ plays a role analogous to conjugation in the complex numbers, and in this analogy we have $\mathrm{Tr}(a) = a + \overline{a}$ and $\mathrm{Nm}(a) = a\overline{a}$. Let $S := \{s \in \mathbb{F}_{q^2} : \mathrm{Tr}(s) = 0\}$, and $S^\times := S \setminus \{0\}$. Note that $S$ is an additive subgroup of $\mathbb{F}_{q^2}$, and $S^\times$ is a coset of $\mathbb{F}_q^\times$ in $\mathbb{F}_{q^2}^\times$. For $\alpha \in \mathbb{F}_{q^2}^\times$, write $S^\times/\alpha := \{s/\alpha : s \in S^\times\}$. The following fact is standard but we will use it a few times so we number it for ease of reference, and give its proof for the reader's convenience.

**Fact 5.2.** *The trace is a surjective $\mathbb{F}_q$-linear map $\mathrm{Tr}\colon \mathbb{F}_{q^2} \to \mathbb{F}_q$.*

*Proof.* Its image lies in $\mathbb{F}_q$ because $\mathrm{Tr}(a)^q = (a + a^q)^q = a^q + a^{q^2} = a^q + a = \mathrm{Tr}(a)$, and $\mathbb{F}_q$ consists of all elements of its algebraic closure that satisfy $x^q = x$. It is $\mathbb{F}_q$-linear because $\mathrm{Tr}(ab) = ab + a^q b^q$, and if $a \in \mathbb{F}_q$, then $a^q = a$, so we get $a^q b^q = ab^q$, and thus $\mathrm{Tr}(ab) = a\,\mathrm{Tr}(b)$ for $a \in \mathbb{F}_q$. Finally, to see it is surjective, we note that as an $\mathbb{F}_q$-linear map to $\mathbb{F}_q$, its image must be 0 or all of $\mathbb{F}_q$, so it suffices to show it is not always zero. To see this, write $\mathbb{F}_{q^2} = \mathbb{F}_q[y]/f(y)$ for some irreducible degree-2 polynomial $f \in \mathbb{F}_q[y]$. Then every element of $\mathbb{F}_{q^2}$ can be written uniquely as $a + by$ for $a, b \in \mathbb{F}_q$. In particular, $y^q = \alpha + \beta y$ for some $\alpha, \beta \in \mathbb{F}_q$. Then we have $\mathrm{Tr}(by) = by + b^q y^q = by + b^q(\alpha + \beta y) = b^q\alpha + y(b + \beta)$. Choosing $b \neq -\beta$ thus gives $\mathrm{Tr}(by) \neq 0$. □

For a subgroup $A \leq \mathbb{F}_{q^2}^\times$, we write $A^{(3)} := \{a^3 : a \in A\}$ for its subgroup of cubes. Unparenthesized powers $A^d$ have their usual meaning as Cartesian powers.

**Convention.** We denote elements of $\mathrm{PSU}_3(q)$ using $3 \times 3$ matrices, which are representatives of their images modulo scalar matrices; to help remind the reader of this we put such matrices in [square brackets] rather than rounded parentheses.

For $x, y \in \mathbb{F}_{q^2}$, define:

$$u(x, y) := \begin{bmatrix} 1 & 0 & 0 \\ y & 1 & 0 \\ x & -y^q & 1 \end{bmatrix}.$$

Now fix a generator $\rho$ of $\mathbb{F}_{q^2}^\times$, and define:

$$h := \begin{bmatrix} \rho & 0 & 0 \\ 0 & \rho^{q-1} & 0 \\ 0 & 0 & \rho^{-q} \end{bmatrix}.$$

Put

$$d := \gcd(3, q+1), \quad M := q^2 - 1, \quad N := \frac{M}{d}.$$

Then $h$ has order $N$ as an element of $\mathrm{PSU}_3(q)$ (that is, modulo scalar matrices): let $\hat{h}$ denote the same matrix but as an element of $\mathrm{SU}_3(q)$. Then $\hat{h}$ has order $M = q^2 - 1$, since $\hat{h}_{11} = \rho$ is a generator of $\mathbb{F}_{q^2}^\times$. We have $h^a = 1$ in $\mathrm{PSU}_3(q)$ iff $\hat{h}^a$ is scalar in $\mathrm{SU}_3(q)$. The latter happens iff $a \equiv a(q-1) \equiv -qa \pmod{q^2-1}$. The fact that the smallest solution to these equations is $a = N = (q^2-1)/gcd(3, q+1)$ is then a straightforward exercise in modular arithmetic.

The subgroup $B = \langle U, h \rangle$ is known as the (or a) *Borel subgroup* of $\mathrm{PSU}_3(q)$, where

$$U = \{u(x,y) : x, y \in \mathbb{F}_{q^2}, x + x^q + y^{q+1} = 0\} = \{u(x,y) : \mathrm{Tr}(x) + \mathrm{Nm}(y) = 0\}.$$

Observe that $h$ normalizes $U$. In particular, $\langle U, h \rangle$ is a solvable group. Furthermore, with the convention $g^a = a^{-1}ga$, we have:

$$u(x,y)^h = u(\lambda x, \beta y), \quad \text{where} \quad \lambda := \rho^{q+1}, \quad \beta := \rho^{2-q}.$$

The element $\lambda$ is primitive in $\mathbb{F}_q^\times$, and $\gcd(M, 2-q) = \gcd(3, q+1) = d$.

We will now describe the structure of $U$. The subgroup $U$ has order $q^3$ and center:

$$Z(U) = \{u(x,0) : x + x^q = 0\} = \{u(s,0) : s \in S\}.$$

We get $|Z(U)| = q$, and $U/Z(U)$ is an elementary Abelian group of order $q^2$. So $Z(U)$ admits an $\mathbb{F}_p$-basis $z_1, \ldots, z_e$, and there exist elements $v_1, \ldots, v_{2e} \in U$ whose images constitute an $\mathbb{F}_p$-basis of $U/Z(U)$. So $\langle U, h \rangle$ is generated by $\{z_1, \ldots, z_e, v_1, \ldots, v_{2e}, h\}$. In particular, every element $g$ of $\langle U, h \rangle$ can be written uniquely as:

$$g = \left( \prod_{i=1}^{e} z_i^{\alpha_i} \right) \cdot \left( \prod_{i=1}^{2e} v_i^{\beta_i} \right) \cdot h^\gamma, \tag{2}$$

where for all $i \in [e]$ and $j \in [2e]$, $0 \le \alpha_i, \beta_j < p$, and $0 \le \gamma < N$. We refer to the expression on the right-hand side of Equation (2), which we can think of as a word in the above generating set, as $\mathrm{dec}(g)$ (for "decomposition").

Now $\mathrm{PSU}_3(q)$ is generated by $\langle U, h \rangle$ and an extra element:

$$t := \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

In particular, $t^2 = 1$ and $h^t = h^{-q}$, so $t$ normalizes $\langle h \rangle$.

Now for any $u = u(x,y) \in U \setminus \{1\}$, there exist unique $g(u), f(u) \in U$ and unique $h(u) \in \langle h \rangle$ such that

$$u(x,y)^t = g(u) \cdot h(u) \cdot t \cdot f(u). \tag{3}$$

Hulpke and Seress gave the following explicit matrix representations for $f(u), g(u), h(u)$, and $u^h$:

$$f(u(x,y)) = \begin{bmatrix} 1 & 0 & 0 \\ -y/x^q & 1 & 0 \\ 1/x & y^q/x & 1 \end{bmatrix}, \quad g(u(x,y)) = \begin{bmatrix} 1 & 0 & 0 \\ -y/x & 1 & 0 \\ 1/x & y^q/x^q & 1 \end{bmatrix}$$

$$h(u(x,y)) = \begin{bmatrix} x & 0 & 0 \\ 0 & x^{q-1} & 0 \\ 0 & 0 & x^{-q} \end{bmatrix}, \quad u(x,y)^h = \begin{bmatrix} 1 & 0 & 0 \\ y\rho^{2-q} & 1 & 0 \\ x\rho^{q+1} & -y^q\rho^{2q-1} & 1 \end{bmatrix}.$$

Furthermore, Hulpke and Seress showed that if $3 \nmid (q+1)$, then $\mathrm{PSU}_3(q)$ admits a short presentation using only 3 relators of the form (3). If $3 \mid (q+1)$, then $\mathrm{PSU}_3(q)$ admits a short presentation using only 7 relators of the form (3). We will recall the precise choices for these relators in Construction 5.7 below.

## 5.2 Uniform presentation for the Borel subgroup of $\mathrm{PSU}_3(q)$

In this section, we prove uniformity of a presentation of the Borel subgroup that is essentially that of [HS01, Lemma 13]; the main difference is that where they merely state that generators with certain properties exist (and for their result any generators with those properties will do), we must construct such generators uniformly. The main result of this section is:

**Lemma 5.3.** *From the standard name (Definition 2.6) for* $\mathrm{PSU}_3(q)$*, in* $\mathsf{DTIME}(\mathrm{polylog}(q))$ *one can construct a presentation of the Borel subgroup* $\langle U, h \rangle$ *that has word length* $O(\log^2 q)$*.*

We start by finding elements (as in the previous section) that are a basis for the elementary Abelian group $Z(U)$, and finding elements whose images are a basis for the elementary Abelian group $U/Z(U)$. Let $\rho \in \mathbb{F}_{q^2}^\times$ be the supplied primitive element in the standard name (Def. 2.6). Define

$$m_\delta := \begin{cases} 0, & p = 2, \\ (q+1)/2, & p \neq 2, \end{cases} \tag{4}$$

and let $\delta := \rho^{m_\delta}$. Then $\delta^q = -\delta$, and hence $S = \ker(\mathrm{Tr}) = \delta\mathbb{F}_q$. Recall that $\lambda = \rho^{q+1}$. For $1 \leq i \leq e$, put $z_i := u(\delta\lambda^{i-1}, 0)$. Since $\lambda$ has degree $e$ over $\mathbb{F}_p$, the elements $1, \lambda, \ldots, \lambda^{e-1}$ form an $\mathbb{F}_p$-basis of $\mathbb{F}_q$. Hence $z_1, \ldots, z_e$ form an $\mathbb{F}_p$-basis of $Z(U)$, and $z_i = z_1^{h^{i-1}}$. We next choose the generators for $U/Z(U)$. Since $\rho$ has degree $2e$ over $\mathbb{F}_p$, the elements $\rho, \rho^2, \ldots, \rho^{2e}$ form an $\mathbb{F}_p$-basis of $\mathbb{F}_{q^2}$. We now compute generators for $U/Z(U)$. To do so, we distinguish the following cases according to $d := \gcd(3, q+1) \in \{1, 3\}$.

- **Case 1:** Suppose that $d = 1$. Using Gaussian elimination over $\mathbb{F}_p$, construct $a_1 \in \mathbb{F}_{q^2}$ satisfying

$$a_1 + a_1^q = -\rho^{q+1},$$

and put $v_1 := u(a_1, \rho)$. Such $a_1$ is guaranteed to exist by Fact 5.2 and the fact that $\rho^{q+1}$ is in $\mathbb{F}_q$. Then $v_1 \in U$, since $\mathrm{Tr}(a_1) + \mathrm{Nm}(\rho) = -\rho^{q+1} + \rho^{q+1} = 0$.

For every integer $b$, $h^{-b} v_1 h^b = u\big(\lambda^b a_1, \rho\beta^b\big)$, so its second parameter is $\rho\beta^b = \rho^{\,1+(2-q)b}$. We have $\rho^{1+(2-q)b} = \rho^t$ precisely when

$$(2-q)b \equiv t - 1 \pmod{M},$$

(where, recall, $M = q^2 - 1$). Since $d = \gcd(M, 2-q) = 1$, for each $1 \le t \le 2e$ there is a unique $b(t) \in \{0, \ldots, M-1\}$ satisfying the above congruence. Define $v_t := h^{-b(t)} v_1 h^{b(t)}$. Then the second parameter of $v_t$ is $\rho^t$. Since $\rho, \rho^2, \ldots, \rho^{2e}$ is an $\mathbb{F}_p$-basis of $\mathbb{F}_{q^2}$, the images of $v_1, \ldots, v_{2e}$ form an $\mathbb{F}_p$-basis of $U/Z(U)$. To compute $b(t)$ from $t$, the inverse of $2-q$ modulo $M$ is computed by the extended Euclidean algorithm in $O(\log q)$ iterations.

- **Case 2:** Suppose that $d = 3$ and $e \ge 2$ (recall $q = p^e$). For $r \in \{1, 2, 3\}$, use Gaussian elimination over $\mathbb{F}_p$ to construct three distinct $a_r \in \mathbb{F}_{q^2}$ satisfying $a_r + a_r^q = -\rho^{r(q+1)}$, which exist by Fact 5.2, and put $v_r := u(a_r, \rho^r)$.

  For $1 \le t \le 2e$, let $r(t) \in \{1, 2, 3\}$ be the unique integer satisfying $r(t) \equiv t \pmod 3$. For every integer $b$, the second parameter of $h^{-b} v_{r(t)} h^b$ is $\rho^{r(t)}\beta^b = \rho^{\,r(t)+(2-q)b}$. Now $\rho^{\,r(t)+(2-q)b} = \rho^t$ precisely when $(2-q)b \equiv t - r(t) \pmod M$. Both sides are divisible by 3, and $\gcd\left(\frac{2-q}{3}, \frac{M}{3}\right) = 1$. Thus the reduced integer congruence

$$\frac{2-q}{3}\, b(t) \equiv \frac{t - r(t)}{3} \pmod N, \quad N := M/3,$$

  has a unique solution $b(t) \in \{0, \ldots, N-1\}$.

  Define $v_t := h^{-b(t)} v_{r(t)} h^{b(t)}$. Then the second parameter of $v_t$ is $\rho^t$, so the images of the $v_t$ form an $\mathbb{F}_p$-basis of $U/Z(U)$.

- **Case 3:** Suppose that $d = 3$, and $e = 1$ (that is, $q = p$). Then $U/Z(U)$ has $\mathbb{F}_p$-dimension two, so a basis cannot contain representatives of all three nonzero $\langle h \rangle$-orbits. For $t \in \{1, 2\}$, use Gaussian elimination to construct two distinct $a_t \in \mathbb{F}_{q^2}$ satisfying $a_t + a_t^q = -\rho^{t(q+1)}$, and put $v_t := u(a_t, \rho^t)$. Such $a_t$ are guaranteed to exist by Fact 5.2 and the fact that $\rho^{q+1}$ is in $\mathbb{F}_q$.

  Since $\rho, \rho^2$ form an $\mathbb{F}_p$-basis of $\mathbb{F}_{q^2}$, the images of $v_1, v_2$ form a basis of $U/Z(U)$.

In order to uniformly compute the relations between our generators, we will need the following uniform method of solving the constructive membership problem in $U$ relative to the generating set we constructed above.

**Lemma 5.4.** *Given $g = u(x, y) \in U$ as a matrix, one can compute, in* DTIME(polylog($q$)), *a word $\mathrm{dec}_U(g) = z_1^{c_1} \cdots z_e^{c_e} v_1^{b_1} \cdots v_{2e}^{b_{2e}}$, $b_i, c_j \in \{0, \ldots, p-1\}$, which evaluates to $g$. Its binary-exponent length is $O(\log q)$.*

*Proof.* The matrix representing $g$ has entries $y = g_{21}$, $x = g_{31}$, (see Section 5.1) so the parameters $x$ and $y$ are read directly from the input matrix. Using Gaussian elimination over $\mathbb{F}_p$, compute the unique coefficients $b_1, \ldots, b_{2e} \in \mathbb{F}_p$ such that $y = \sum_{t=1}^{2e} b_t \rho^t$. Compute, in the indicated order, the matrix $W := \prod_{t=1}^{2e} v_t^{b_t}$. The second parameter is additive under multiplication in $U$, so the second parameter of $W$ is $\sum_{t=1}^{2e} b_t \rho^t = y$. Consequently, $gW^{-1} = u(s, 0)$ for some $s \in S$. The

element $s$ is read from the $(3,1)$-entry of the matrix $gW^{-1}$. Using Gaussian elimination again, compute the unique coefficients $c_1, \dots, c_e \in \mathbb{F}_p$ satisfying $s = \sum_{i=1}^{e} c_i \delta \lambda^{i-1}$. It follows that $g = \left(\prod_{i=1}^{e} z_i^{c_i}\right)\left(\prod_{t=1}^{2e} v_t^{b_t}\right)$. All linear systems have dimension $O(e)$ over $\mathbb{F}_p$, and all powers are evaluated by repeated squaring. Hence the word is computed in time polylog$(q)$ and has binary-exponent length $O(e \log p) = O(\log q)$. □

For $g \in U$, let $\widehat{\mathrm{dec}}_U(g)$ denote the same word with the matrix generators replaced by abstract symbols.

**Construction 5.5** (Uniform presentation of the Borel subgroup)**.** Let $\mathcal{P}_B$ have generators $\widehat{z}_1, \dots, \widehat{z}_e$, $\widehat{v}_1, \dots, \widehat{v}_{2e}$, $\widehat{h}$. We add the following relations so that $\widehat{z}_1, \dots, \widehat{z}_e$ generate $Z(U)$; $\widehat{v}_1, \dots, \widehat{v}_{2e}$ are lifts of a basis of $U/Z(U)$; and $\widehat{h}$ normalizes the subgroup generated by the $\widehat{z}_i$ and $\widehat{v}_t$:

1. Power and order relations.
   - For $1 \le i \le e$, add $\widehat{z}_i^{\,p} = 1$.
   - For $1 \le t \le 2e$, add $\widehat{v}_t^{\,p} = \widehat{\mathrm{dec}}_U(v_t^p)$. Since $U/Z(U)$ has exponent $p$, we have $v_t^p \in Z(U)$. More explicitly, $v_t^p = 1$ in odd characteristic, while $v_t^2 \in Z(U)$ in characteristic two.[25] Consequently, $\widehat{\mathrm{dec}}_U(v_t^p)$ is a word only in $\widehat{z}_1, \dots, \widehat{z}_e$.
   - Add $\widehat{h}^{\,N} = 1$.
2. ($\widehat{z}_1$ is central) Add $[\widehat{z}_1, \widehat{z}_i] = 1 \quad (1 \le i \le e)$, $[\widehat{z}_1, \widehat{v}_t] = 1 \quad (1 \le t \le 2e)$.
3. Add the following quotient-commutator relations. Since $[U, U] \le Z(U)$, every right-hand side below is a word only in the $\widehat{z}_i$.
   - If $d = 1$, add $[\widehat{v}_1, \widehat{v}_t] = \widehat{\mathrm{dec}}_U([v_1, v_t])$ $(2 \le t \le 2e)$.
   - If $d = 3$ and $e \ge 2$, add $[\widehat{v}_r, \widehat{v}_t] = \widehat{\mathrm{dec}}_U([v_r, v_t])$ for $r \in \{1,2,3\}$ and $1 \le t \le 2e$.
   - If $d = 3$ and $e = 1$, add $[\widehat{v}_1, \widehat{v}_2] = \widehat{\mathrm{dec}}_U([v_1, v_2])$.
4. For every $1 \le i \le e$, add $\widehat{h}^{-1}\widehat{z}_i\widehat{h} = \widehat{\mathrm{dec}}_U(z_i^h)$, and for every $1 \le t \le 2e$, add $\widehat{h}^{-1}\widehat{v}_t\widehat{h} = \widehat{\mathrm{dec}}_U(v_t^h)$.
5. Add the orbit relations $\widehat{z}_i = \widehat{h}^{-(i-1)}\widehat{z}_1\widehat{h}^{\,i-1}$ $(2 \le i \le e)$. If $d = 1$, also add $\widehat{v}_t = \widehat{h}^{-b(t)}\widehat{v}_1\widehat{h}^{\,b(t)}$ $(2 \le t \le 2e)$. If $d = 3$ and $e \ge 2$, also add $\widehat{v}_t = \widehat{h}^{-b(t)}\widehat{v}_{r(t)}\widehat{h}^{\,b(t)}$ $\left(t \notin \{1,2,3\}\right)$.

We now have all the ingredients for a uniform presentation of the Borel subgroup:

*Proof of Lemma 5.3.* All displayed relations hold for the indicated matrices, so there is a surjective homomorphism from the presented group onto $B$.

Let $\widehat{U}$ be the subgroup generated by the $\widehat{z}_i$ and $\widehat{v}_t$. The conjugation relations and $\widehat{h}^N = 1$ imply that $\widehat{h}$ normalizes $\widehat{U}$. Since every $\widehat{z}_i$ is an $\widehat{h}$-conjugate of $\widehat{z}_1$, and $\widehat{z}_1$ commutes with every generator of $\widehat{U}$, all the $\widehat{z}_i$ are central in $\widehat{U}$.

Suppose first that $d = 1$. Every $\widehat{v}_i$ is an $\widehat{h}$-conjugate of $\widehat{v}_1$. Conjugating the commutator relations and expanding commutators of products expresses every $[\widehat{v}_i, \widehat{v}_j]$ as a word in the central generators $\widehat{z}_1, \dots, \widehat{z}_e$. If $d = 3$ and $e \ge 2$, the same argument uses the three representatives $\widehat{v}_1, \widehat{v}_2, \widehat{v}_3$. If $d = 3$ and $e = 1$, the only quotient commutator relation was included directly.

[25] In characteristic 2, the quotient-basis lifts $v_i$ may have order 4, so we use the correct power relations $\widehat{v}_t^2 = \widehat{\mathrm{dec}}_U\left(v_t^2\right), , v_t^2 \in Z(U)$ rather than asserting that these lifts always have order $p$.

The power relations now let us collect every word in $\widehat{U}$ to the form $\widehat{z}_1^{c_1}\cdots\widehat{z}_e^{c_e}\widehat{v}_1^{b_1}\cdots\widehat{v}_{2e}^{b_{2e}}$ with $0 \le b_i, c_j < p$. Hence $|\widehat{U}| \le p^{3e} = q^3$. The conjugation relations let us move every occurrence of $\widehat{h}$ to the right, so the whole presented group has order at most $q^3 N = |B|$. The surjection onto $B$ is thus an isomorphism.

We now account for the length. There are $O(e)$ relations. Each word has binary-exponent length $O(\log q)$, and every orbit exponent has $O(\log q)$ bits. Thus the total binary-exponent length is $O(e \log q) \subseteq O(\log^2 q)$. By Lemma 5.4, every right-hand side is computed using field arithmetic, Gaussian elimination, and the Euclidean algorithm in time polynomial in $\log q$. By Remark 2.15 this can be uniformly converted into a presentation with the asymptotically the same word-length. □

### 5.3 Proof of Proposition 5.1

To complete the uniform presentation of $\mathrm{PSU}_3(q)$, we now make the three and seven additional relations of [HS01, Theorems 15–16] uniform. We choose their first coordinates to be explicitly known powers of $\rho$.

Recall several parameters from above: $M = q^2 - 1$, $\delta = \rho^{m_\delta}$, where $m_\delta$ is defined in (4). Let

$$\varepsilon_- := \begin{cases} 0, & p = 2, \\ M/2, & p \neq 2, \end{cases}$$

so that $-1 = \rho^{\varepsilon_-}$. Put

$$x := \delta\rho,\ \ s := -\delta(\rho + \rho^q),\ \ x' := x + s = -\delta\rho^q.$$

The element $\rho + \rho^q$ belongs to $\mathbb{F}_q$ and is nonzero. Indeed, in characteristic two, equality $\rho + \rho^q = 0$ would put $\rho$ in $\mathbb{F}_q$; in odd characteristic it would imply $\rho^{q-1} = -1$, so the order of $\rho$ would divide $2(q-1) < q^2 - 1$. Hence $s \in S^\times$. Moreover, $x \in \rho S^\times$ and $x' = -\delta\rho^q = \delta\rho^{-1}(-\rho^{q+1}) \in S^\times/\rho$. Here, $S^\times/\rho$ denotes the set $\{s/\rho : s \in S^\times\}$. The relevant first coordinates are known powers: $\delta = \rho^{m_\delta}$, $x = \rho^{m_\delta+1}$, $x' = \rho^{m_\delta+q+\varepsilon_-}$.

Set $c := u(\delta, 0) = z_1$, $b := u(s, 0)$. Set $r_0 := -\operatorname{Tr}(x) \in \mathbb{F}_q^\times$. Indeed, $x \notin S$, so $\operatorname{Tr}(x) \neq 0$. Since $s \in S$ and $x' = x + s$, we also have $-\operatorname{Tr}(x') = r_0$.

Below we use the standard deterministic square-root algorithm for finite fields of odd order with a supplied quadratic nonresidue: given a known nonsquare in $\mathbb{F}_q^\times$, quadratic residuosity testing and square-root extraction for squares are computable in time polynomial in $\log q$; see [AMM77]. When $p$ is odd, the primitive element $\lambda = \rho^{q+1}$ is such a known nonsquare.

**Lemma 5.6.** *Given $r \in \mathbb{F}_q^\times$, one can compute, in deterministic time polynomial in $\log q$, an element $L(r) \in \mathbb{F}_{q^2}^\times$ such that $\mathrm{Nm}(L(r)) = r$. If $d = 3$, one can moreover compute such an element $L_3(r)$ satisfying $L_3(r) \in (\mathbb{F}_{q^2}^\times)^{(3)}$.*

*Proof.* Suppose first that $p = 2$. Put $L(r) := r^{2^{e-1}} \in \mathbb{F}_q$ (recall $e = \log_p q$). Then $L(r)^2 = r$. Since $L(r) \in \mathbb{F}_q$, its relative norm is $\mathrm{Nm}_{\mathbb{F}_{q^2}/\mathbb{F}_q}(L(r)) = L(r)^{q+1} = L(r)^2 = r$. If $d = 3$, then $3 \mid (q+1)$, and hence $\mathbb{F}_q^\times = \langle \rho^{q+1} \rangle \subseteq (\mathbb{F}_{q^2}^\times)^{(3)}$. Thus $L_3(r) := L(r)$ is already a cube in $\mathbb{F}_{q^2}^\times$.

Now suppose instead that $p$ is odd. The element $\lambda = \rho^{q+1}$ is primitive, and hence nonsquare, in $\mathbb{F}_q^\times$. Using the deterministic square-root algorithm with this known nonsquare [AMM77], we proceed as follows.

If $r$ is a square, choose $a \in \mathbb{F}_q^\times$ satisfying $a^2 = r$ and put $L(r) := a$. Then we have $\mathrm{Nm}(a) = a^2 = r$, as desired. Next, if instead $r$ is a nonsquare, then $r/\lambda$ is a square, because it is the quotient of two nonsquares. In this case, choose $a \in \mathbb{F}_q^\times$ satisfying $a^2 = r/\lambda$ and put $L(r) := \rho a$. Now we also have $\mathrm{Nm}(\rho a) = \lambda a^2 = r$, as desired.

Finally, suppose additionally that $d = 3$. Since $3 \mid (q+1)$, every element of $\mathbb{F}_q^\times$ is a cube in $\mathbb{F}_{q^2}^\times$. If $r$ is a square, choose $a^2 = r$ and put $L_3(r) := a$ as above. If instead $r$ is a nonsquare, then $\lambda^3$ is also a nonsquare, because $\lambda$ is primitive and 3 is odd. Hence $r/\lambda^3$ is a square. Choose $a \in \mathbb{F}_q^\times$ satisfying $a^2 = r/\lambda^3$ and put $L_3(r) := \rho^3 a$. Here $a$ is a cube in $\mathbb{F}_{q^2}^\times$, and therefore $\rho^3 a$ is a cube. Moreover, $\mathrm{Nm}(\rho^3 a) = \lambda^3 a^2 = r$. All of these operations take time polynomial in $\log q$. □

For the presentation of $\mathrm{PSU}_3(q)$ [HS01], we recall one additional type of relator. For a given element $u \in U$, define the relation:

$$\mathcal{R}(u): \ \widehat{t}\,\widehat{\mathrm{dec}}_U(u)\,\widehat{t} = \widehat{\mathrm{dec}}_U(g(u)) \cdot \widehat{h}^{m(u)} \cdot \widehat{t} \cdot \widehat{\mathrm{dec}}_U(f(u)), \tag{5}$$

where $m(u)$ is the explicitly known exponent of the first coordinate of $u$.

**Construction 5.7.** Start with $\mathcal{P}_B$ (from Construction 5.5) and add the generator $\widehat{t}$. We now add the following relators:

- $\widehat{t}^2 = 1$.
- $\widehat{t}\widehat{h}\widehat{t} = \widehat{h}^{-q}$
- We now add relators depending on the value of $d$ (1 or 3).
  - **Case 1:** $d = 1$. Let $\nu := L(r_0)$ (from Lemma 5.6), $a := u(x, \nu)$, $a' := u(x', \nu) = ab$ (where $x, x'$ were defined at the top of Section 5.3). Add the three relations $\mathcal{R}(a)$, $\mathcal{R}(c)$, $\mathcal{R}(a')$ using (5). Here $m(a) = m_\delta + 1$, $m(c) = m_\delta$, $m(a') = m_\delta + q + \varepsilon_-$ modulo $M$.
  - **Case 2:** $d = 3$. let $\nu_0 := L_3(r_0)$, $\kappa := \rho^{q-1}$, $\nu_j := \nu_0 \kappa^j$ $(j = 0, 1, 2)$. Then $\mathrm{Nm}(\kappa) = 1$, and, since $q - 1 \equiv 1 \pmod 3$, the elements $\nu_0, \nu_1, \nu_2$ lie in the three distinct cosets of $(\mathbb{F}_{q^2}^\times)^{(3)}$ in $\mathbb{F}_{q^2}^\times$. Put $a_j := u(x, \nu_j)$, $a'_j := u(x', \nu_j) = a_j b$ $(j = 0, 1, 2)$.
    Add the seven relations $\mathcal{R}(c)$ and $\mathcal{R}(a_j)$, $\mathcal{R}(a'_j)$ $(j = 0, 1, 2)$. The corresponding exponents are $m(a_j) = m_\delta + 1$, $m(c) = m_\delta$, $m(a'_j) = m_\delta + q + \varepsilon_-$ modulo $M$.

**Lemma 5.8.** *The presentation of Construction 5.7 presents* $\mathrm{PSU}_3(q)$.

*Proof.* Since $\mathcal{P}_B$ presents $B$, the arguments of [HS01, Lemmas 5, 7, and 14 and Theorems 15–16] apply with our normal-form words $\widehat{\mathrm{dec}}_U$ in place of theirs. The indicated matrices satisfy all defining relations and generate $\mathrm{PSU}_3(q)$, so there is an epimorphism from the presented group to $\mathrm{PSU}_3(q)$.

The elements $b = u(s, 0)$ and $c = u(\delta, 0)$ are nonidentity elements of $Z(U)$. By [HS01, Lemma 8], they are conjugate under $\langle h \rangle$. By [HS01, Lemma 7], the relation $\mathcal{R}(c)$ therefore implies the relation $\mathcal{R}(b)$ in the abstract presentation.

If $d = 1$, the presentation consequently implies $\mathcal{R}(a)$, $\mathcal{R}(b)$, $\mathcal{R}(ab)$. The hypotheses of [HS01, Theorem 15] hold because $x \in \rho S^\times$, $s \in S^\times$, and $x + s = x' \in S^\times/\rho$. Hence the presentation defines $\mathrm{PSU}_3(q)$.

If $d = 3$, the elements $\nu_0, \nu_1, \nu_2$ have the same norm and represent the three cube classes. The presentation therefore implies $\mathcal{R}(a_0 b), \mathcal{R}(a_1 b), \mathcal{R}(a_2 b)$. The hypotheses of [HS01, Theorem 16] hold, and the same conclusion follows. □

Finally, we have essentially completed the proof of Proposition 5.1, the uniform short presentation of $\mathrm{PSU}_3(q)$; here we gather the last remaining pieces of the size and complexity analysis, to complete the proof:

*Proof of Proposition 5.1.* All field elements are represented as polynomials of degree less than $2e$ over $\mathbb{F}_p$. Addition, multiplication, inversion, powering, trace, and norm are computable in time polynomial in $e \log p = \log q$. The trace equations and all coordinate expansions are solved by Gaussian elimination on vector spaces of dimension $O(e)$ over $\mathbb{F}_p$. The exponents $e_t$ are computed by the Euclidean algorithm on $O(\log q)$-bit integers. Lemma 5.6 uses only exponentiation and square-root extraction with the explicitly known nonsquare $\lambda = \rho^{q+1}$.

By Lemma 5.3, the Borel part has binary-exponent length $O(\log^2 q)$. The Bruhat part adds only three or seven relations, each of binary-exponent length $O(\log q)$. Thus the complete presentation has binary-exponent length $O(\log^2 q)$.

Finally, apply the repeated-squaring conversion of [BGK+97, Remark 1.3] (reproduced in Remark 2.15 above) simultaneously to every exponent in the presentation. This converts the presentation to the word-length convention used in this paper while preserving the $O(\log^2 q)$ bound, and the conversion is itself computable in time polynomial in $\log q$. Since $|\mathrm{PSU}_3(q)| = \frac{q^3(q^3+1)(q^2-1)}{\gcd(3,q+1)}$, we have $\log|G| = \Theta(\log q)$. The resulting presentation has word length $O(\log^2|G|)$ and is produced in $\mathsf{DTIME}(\mathrm{polylog}(n))$. □

# 6 Uniformity of the Guralnick–Kantor–Kassabov–Lubotzky Short Presentation for the Suzuki groups ${}^2B_2(q) = \mathrm{Sz}(q)$

Hulpke and Seress [HS01, p. 720] write that J. Thompson told W. Kantor in personal communication that the original paper of Suzuki [Suz62] contains within it, at least implicitly, a short presentation of the Suzuki groups ${}^2B_2(q) = \mathrm{Sz}(q)$. Guralnick, Kantor, Kassabov, and Lubotzky [GKKL08] gave a short and bounded[26] presentation of these groups, but do not say anything about uniformity. We show that their presentation can be made uniform. Precisely, we will establish the following.

**Proposition 6.1.** *Let $k > 0$, and let $q = 2^{2k+1}$. There exists a $\mathsf{DTIME}(\mathrm{polylog}(q))$ algorithm that, given the standard name for $\mathrm{Sz}(q)$, constructs a short presentation for $\mathrm{Sz}(q)$.*

## 6.1 Preliminaries: polynomial notation

If $\gamma$ lies in an extension field of $\mathbb{F}_p$, then $m_\gamma(x)$ denotes its minimal polynomial over $\mathbb{F}_p$. If $\delta \in \mathbb{F}_p[\gamma]$, let $f_{\delta;\gamma}(x) \in \mathbb{F}_p[x]$ denote the unique polynomial satisfying $f_{\delta;\gamma}(\gamma) = \delta$, $\deg(f_{\delta;\gamma}) < \deg(m_\gamma)$.

Following Guralnick, Kantor, Kassabov, and Lubotzky's polynomial notation [GKKL08, Section 4.3.0, Equations (4.15)–(4.16)], for any polynomial $g(x) = \sum_{i=0}^{e} g_i x^i \in \mathbb{Z}[x]$ and any two elements $u, h$ in a group, define $[[u^{g(x)}]]_h := \prod_{i=0}^{e} (u^{g_i})^{h^i}$, where the product is taken in increasing order of $i$, and we are using the convention $x^h = h^{-1}xh$. Equivalently,

$$[[u^{g(x)}]]_h = u^{g_0} h^{-1} u^{g_1} h^{-1} \cdots h^{-1} u^{g_e} h^e.$$

[26] In the context of the Short Presentation Conjecture, whereas "short" means polylogarithmic length, "bounded" means $O(1)$ generators and $O(1)$ relators.

We spell out the following effective procedures related to the above because they will be useful for establishing uniformity of the presentation below.

**Lemma 6.2.** *Let $q = p^e$, and suppose the standard name gives $\mathbb{F}_q$ over $\mathbb{F}_p$ together with a primitive element $\zeta \in \mathbb{F}_q^\times$. Then the following can be done in* $\mathsf{DTIME}(\mathrm{polylog}(q))$.

1. *Compute the minimal polynomial $m_\zeta(x)$.*
2. *Given $\delta \in \mathbb{F}_q$, compute the "coordinate polynomial" $f_{\delta;\zeta}(x)$ satisfying $f_{\delta;\zeta}(\zeta) = \delta$ and $\deg(f_{\delta;\zeta}) < e$.*
3. *For every $s \le O(e)$, compute $x^{p^s} \bmod m_\zeta(x)$.*
4. *Given $g(x) = \sum_i g_i x^i$ with $\deg(g) \le e$ and $0 \le g_i < p$, output the word $[[u^{g(x)}]]_h$ (as an element of a free group in which $u, h$ are two of the generators). For $p = 2$, this word has length $O(e)$.*

*Proof.* 1. The standard name realizes $\mathbb{F}_q$ as $\mathbb{F}_p[Y]/f(Y)$. Compute the unique linear dependence among $1, \zeta, \zeta^2, \ldots, \zeta^e$ over $\mathbb{F}_p$. This is Gaussian elimination on an $(e+1) \times e$ matrix over $\mathbb{F}_p$, hence takes time polynomial in $e$ and $\log p$. (Since $\zeta$ is a primitive element, we have that $1, \zeta, \ldots, \zeta^{e-1}$ are linearly independent, for otherwise the field $\mathbb{F}_p(\zeta)$ would be smaller than order $q$.)

2. Express $\delta$ in the basis $1, \zeta, \ldots, \zeta^{e-1}$ over $\mathbb{F}_p$ by solving a linear system. The resulting coefficients are the coefficients of $f_{\delta;\zeta}(x)$.

3. Compute $R_0(x) = x$, $R_{i+1}(x) = R_i(x)^p \bmod m_\zeta(x)$. After $s$ iterations, $R_s(x) = x^{p^s} \bmod m_\zeta(x)$. Since all polynomials have degree $< e$, this takes $\mathrm{poly}(e, \log p) = \mathrm{polylog}(q)$ time.

4. When $p = 2$, each coefficient is 0 or 1, so one directly expands $[[u^{g(x)}]]_h$ omitting the factors $u^0$. This has length $O(e)$. □

## 6.2 Preliminaries on the Suzuki groups

Let $q = 2^{2k+1}$. We follow the notation of [GKKL08] for ease of reference and comparison, and this material can be found there with further references. Throughout this section, we will assume $q > 8$. For if $q \le 8$, we can provide the multiplication table. Let $\mathbb{F}_q^\times = \langle \zeta \rangle$, and let $\theta : x \mapsto x^{2^{k+1}}$ be the field automorphism whose square is the Frobenius automorphism $x \mapsto x^2 = (x^\theta)^\theta$. Define $U$ to be the group consisting of all pairs $(\alpha, \beta) \in \mathbb{F}_q^2$ with the multiplication rule:

$$(\alpha, \beta) \cdot (\gamma, \delta) = (\alpha + \gamma, \beta + \delta + \alpha\gamma^\theta),$$

and $W := Z(U) = [U, U] = \{(0, \beta) : \beta \in \mathbb{F}_q\}$.

For $\epsilon \in \mathbb{F}_q^\times$, let $h_\epsilon$ denote the automorphism of $U$ defined by

$$(\alpha, \beta)^{h_\epsilon} = (\epsilon\alpha, \epsilon^{\theta+1}\beta). \tag{6}$$

Then $B = U \rtimes \langle h_\zeta \rangle$ is isomorphic to the Borel subgroup of $\mathrm{Sz}(q)$. The cyclic group $\langle h_\zeta \rangle \cong \mathbb{F}_q^\times$ is transitive on the non-trivial elements of both $Z(U)$ and $U/Z(U)$. If $x \in U$ and $v = (i, j, k, \ell) \in \mathbb{Z}^4$, we write:

$$x^v = x^{(i,j,k,\ell)} = x^h \text{ where } h = h_\zeta^i h_{\zeta^\theta}^j h_{\zeta+1}^k h_{\zeta^\theta+1}^\ell \tag{7}$$

[GKKL08, Equation (4.28), p. 29]. We will need the following set $S_0$ of 16 vectors from $\mathbb{Z}^4$ [GKKL08, Table B.4].

$$\begin{aligned} S_0 = \{&(-1,-1,0,0),(-1,0,0,0),(0,-1,0,0),(0,0,-1,-1),(0,0,-1,0)\\ &(0,0,0,-1),(0,0,0,0),(0,1,0,0),(1,0,0,0),(1,0,1,0),\\ &(1,1,0,0),(1,1,1,0),(1,1,1,1),(2,1,1,1)\\ &(-2,-2,0,0),(-2,-1,0,0)\}. \end{aligned}$$

With this notation in hand, we recall the short, bounded presentation for $\mathrm{Sz}(q)$ from [GKKL08]. We note that in this presentation, the elements $h_\zeta$ etc. will line up, under isomorphism, with the elements $h_\zeta$ described above, but they are not *assumed* to do so; that is, in the presentation, the elements $h_\zeta, h_{\zeta+1}, h_{\zeta^\theta} h_{\zeta^\theta+1}$ are merely four independent generators, and the relations between them are described explicitly by the relators of the presentation (there are no hidden implicit relations implied by the notation).

**Theorem 6.3** ([GKKL08, Section 4.4.3])**.** *Let $k > 0$, and let $q = 2^{2k+1}$. $\mathrm{Sz}(q)$ admits the following presentation with 7 generators and 43 relations.*

***Generators:*** $u, w, t, h_\zeta, h_{\zeta+1}, h_{\zeta^\theta}, h_{\zeta^\theta+1}$.

***Relations:*** *In the relations below, $a = h_\zeta^{-1} h_{\zeta^\theta}$, $w_1, \ldots, w_5$ are suitable elements of $W := \langle w^{\langle a \rangle} \rangle$ and $u_1, u_2, u_3, u_{\star,1}, u_{\star,2}, u_{\star,3}$ are suitable elements of $U := \langle u^{\langle h_\zeta \rangle} \rangle$, determined by our choice of $u$ and $\zeta$. Here and below, the notation $[[x^{g(x)}]]_h$ is the polynomial word notation from Section 6.1.*

1. $[h_\star, h_\bullet] = 1$ *for all* $\star, \bullet \in \{\zeta, \zeta^\theta, \zeta+1, \zeta^\theta+1\}$.
2. $w = u^2$ *and* $w^2 = 1$.
3. $w^{(0,0,-1,1)} = w w^{(-1,1,0,0)}$ *(see (7) above for notation).*
4. $w^{(0,0,2,-1)} = w w^{(2,-1,0,0)}$.
5. $[w, w^{(-1,1,0,0)}] = 1$.
6. $[[w^{m_\zeta(x)}]]_a = 1$.
7. $w^{(1,0,0,0)} = [[w^{x^{2^{k+1}+1}}]]_a$, *where $x^{2^{k+1}+1}$ is reduced modulo $m_\zeta(x)$ in order to obtain a short word.*
8. $u^{(1,0,0,0)} = u u^{(0,0,1,0)} w_1$.
9. $u^{(0,1,0,0)} = u u^{(0,0,0,1)} w_2$.
10. $[u^v, w] = 1$ *for every* $v \in S_0$.
11. $[u, u^{(1,0,0,0)}] = w_3$.
12. $[[u^{m_\zeta(x)}]]_{h_\zeta} = w_4$.
13. $u^{(0,1,0,0)} = [[u^{x^{2^{k+1}}}]]_{h_\zeta} \cdot w_5$, *where $x^{2^{k+1}}$ is reduced modulo $m_\zeta(x)$ in order to obtain a short word.*

14. $t^2 = 1$.

15. $h_\star^t = h_\star^{-1}$ *for all* $\star \in \{\zeta, \zeta+1, \zeta^\theta, \zeta^\theta+1\}$.

16. $t = u_1 u_2^t u_3$.

17. $h_\star t = u_{\star,1} u_{\star,2}^t u_{\star,3}$ *for all* $\star \in \{\zeta, \zeta+1, \zeta^\theta, \zeta^\theta+1\}$.

Relations (1)–(13) are the Borel relations from [GKKL08, Proposition 4.30], and relations (14)–(17) are the additional relations used in [GKKL08, Proposition 4.30 and Section 4.4.3]. The presentation itself and its correctness are due to [GKKL08]. What we add below is the uniformity, by showing that all words appearing in the presentation can be constructed from the standard name in time $\mathsf{DTIME}(\mathrm{polylog}(q))$.

### 6.3 Proof of Proposition 6.1

As in the cases of the Ree groups (Lemma 3.5) and $\mathrm{PSU}_3(q)$ (Lemma 5.4), another key to our approach to a uniform presentation here is the following lemma, which solves the constructive membership problem in $U$ relative to a particular generating set.

**Convention.** Throughout the remainder of this section we use the notation $U(A, B)$ to denote the element of $U$ that has coordinates $(A, B)$ in the notation from the start of Section 6.2.

**Lemma 6.4.** *Let* $u = U(1,0)$ *and* $w = U(0,1)$*, and let* $h_\zeta, h_{\zeta^\theta}$ *be as above. There is a* $\mathsf{DTIME}(\mathrm{polylog}(q))$ *algorithm which, given* $(A, B) \in \mathbb{F}_q^2$ *as input, outputs a word* $\mathrm{Word}_{A,B}$ *over four generators* $\hat{u}, \hat{w}, \hat{h}_\zeta, \hat{h}_{\zeta^\theta}$ *such that* $\mathrm{Word}_{A,B}(u, w, h_\zeta, h_{\zeta^\theta}) = U(A, B)$.

*Proof.* Let $a = h_\zeta^{-1} h_{\zeta^\theta}$ and let $\hat{a}$ be the corresponding element of the free group, viz. $\hat{a} = \hat{h}_\zeta^{-1} \hat{h}_{\zeta^\theta}$. By the action formula (6), the element $a$ acts on $W = Z(U)$ as multiplication by $\zeta$. Indeed, $(\zeta^{-1}\zeta^\theta)^{\theta+1} = \zeta^{\theta^2-1} = \zeta$. Hence, if $D \in \mathbb{F}_q$ and $f_{D;\zeta}(x) = \sum_{i=0}^{e-1} d_i x^i$ is the coordinate polynomial satisfying $f_{D;\zeta}(\zeta) = D$, then $[[\hat{w}^{f_{D;\zeta}(x)}]]_{\hat{a}}$ is a word of length $O(e) = O(\log q)$ evaluating to $U(0, D)$.

Similarly, for any $A \in \mathbb{F}_q$, if $f_{A;\zeta}(x) = \sum_{i=0}^{e-1} a_i x^i$, then $P_A := [[\hat{u}^{f_{A;\zeta}(x)}]]_{\hat{h}_\zeta}$ has first coordinate $A$. Say $P_A = U(A, C_A)$, where $C_A$ is computed by simulating the multiplication rule in $U$. Explicitly, $C_A = \sum_{0 \le i < j < e} a_i a_j \, \zeta^i (\zeta^j)^\theta$. Therefore, for any $A, B \in \mathbb{F}_q$, the word

$$\mathrm{Word}_{A,B} := [[\hat{u}^{f_{A;\zeta}(x)}]]_{\hat{h}_\zeta} \, [[\hat{w}^{f_{B-C_A;\zeta}(x)}]]_{\hat{a}}$$

evaluates to $U(A, B)$. By Lemma 6.2, this word is computable in $\mathsf{DTIME}(\mathrm{polylog}(q))$ and has length $O(e) = O(\log q)$. □

We how have all the ingredients in place to prove Proposition 6.1.

*Proof of Proposition 6.1.* Let $e = 2k+1 = \log_2 q$. We construct the presentation from Theorem 6.3. The correctness of this presentation was established in [GKKL08, Section 4.4.3]. It remains only to show that the relators can be output in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

We now show the relators can be generated in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

(1)–(5), (10), (14)–(15) Each of these relators is fixed (involves only fixed constants independent of $q$ and have constant word-length). To see this for relation (10), note that (10) uses the fixed set $S_0$ of 16 vectors (see Section 6.2). Whenever a word of the form $x^{(i,j,r,s)}$ occurs, we expand it as $x^{h_\zeta^i h_{\zeta^\theta}^j h_{\zeta+1}^r h_{\zeta^\theta+1}^s}$. All vectors appearing here are fixed, so these relators can be written down in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

(6) By Lemma 6.2, we compute $m_\zeta(x)$ from the standard name in $\mathsf{DTIME}(\mathrm{polylog}(n))$. Since $p = 2$, all coefficients of $m_\zeta(x)$ are 0 or 1. Thus

$$[[w^{m_\zeta(x)}]]_a$$

expands directly as a word of length $O(e) = O(\log q)$ in $w$ and $a$. Since $a = h_\zeta^{-1} h_{\zeta^\theta}$ has constant length in the generators from Theorem 6.3, relation (6) can be output in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

(7) Again by Lemma 6.2, we compute

$$g_{\theta+1}(x) := x^{2^{k+1}+1} \bmod m_\zeta(x)$$

in $\mathsf{DTIME}(\mathrm{polylog}(n))$. The polynomial $g_{\theta+1}(x)$ has degree $< e$ and coefficients in $\mathbb{F}_2$, so $[[w^{g_{\theta+1}(x)}]]_a$ has length $O(e) = O(\log q)$ and, again by Lemma 6.2, can be computed efficiently. Hence relation (7) can be output in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

(8)–(9), (11) The elements $w_1, w_2, w_3$ are the central correction terms appearing in [GKKL08, Proposition 4.30]. In the explicit pair model for $U$ from the start of Section 6.2, compute

$$D_1 := (u\, u^{(0,0,1,0)})^{-1} u^{(1,0,0,0)} \qquad D_2 := (u\, u^{(0,0,0,1)})^{-1} u^{(0,1,0,0)} \qquad D_3 := [u, u^{(1,0,0,0)}].$$

By [GKKL08], these elements lie in $W = Z(U)$. Define $B_i$ by $D_i = U(0, B_i)$ $(i = 1, 2, 3)$. Then output

$$w_i := \mathrm{Word}_{0,B_i}.$$

from Lemma 6.4. This gives words of length $O(e) = O(\log q)$ for $w_1, w_2, w_3$, computable in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

(12)–(13) For relation (12), compute

$$D_4 := [[u^{m_\zeta(x)}]]_{h_\zeta}.$$

By relator (12), this element is the central correction $w_4$. Define $B_4$ by $D_4 = U(0, B_4)$ and output $w_4 := \mathrm{Word}_{0,B_4}$ from Lemma 6.4. For relation (13), compute

$$g_\theta(x) := x^{2^{k+1}} \bmod m_\zeta(x)$$

using Lemma 6.2, and then compute

$$D_5 := \left([[u^{g_\theta(x)}]]_{h_\zeta}\right)^{-1} u^{(0,1,0,0)}.$$

By relator (13), this is the central correction $w_5$. Define $B_5$ by $D_5 = U(0, B_5)$ and output

$$w_5 := \mathrm{Word}_{0,B_5}.$$

Thus relations (12) and (13) can thus be output in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

(16)–(17) We now use Suzuki's explicit formulas. Guralnick, Kantor, Kassabov, and Lubotzky [GKKL08, Section 4.4.3] state that relation (16) is due to Suzuki's [Suz62, Formula (13)], and that explicit versions of relations (16) and (17) are given by Suzuki's [Suz62, Formulas (13) and (38)]. We use these formulas in the notation of the present section.

To avoid conflict with Suzuki's element $\rho$, put $\varrho := 2^k$. Then $\varrho\theta = q$ as exponents on $\mathbb{F}_q$. Suzuki's formula (38), translated into the present notation, gives, for every $\lambda \in \mathbb{F}_q^\times$,

$$h(\lambda)t = U(\lambda^{-\varrho}, 0)\, U(0, \lambda^{\varrho+1})^t\, U(\lambda^{-\varrho}, \lambda^{-(\varrho+1)}). \tag{8}$$

Equivalently, this identity follows by direct multiplication in Suzuki's $4 \times 4$ matrix model, see Suzuki's formula (38) [Suz62, (38)].

For relation (16), take $\lambda = 1$ in (8). Then $t = U(1,0)U(0,1)^t U(1,1)$. Suzuki also notes in the paragraph preceding [Suz62, Formula (39)] that one may take $\rho = (1,0)$ and $\sigma = (0,1)$ [Suz62, paragraph preceding formula (39)]. Thus in our notation $\rho = u$ and $\sigma = w = u^2$. Since

$$U(1,1) = U(1,0)U(0,1) = uw,$$

we may choose

$$u_1 = u, \qquad u_2 = w, \qquad u_3 = uw.$$

Hence relation (16) is output as $t = u\, w^t\, u\, w$.

For relation (17), let

$$\lambda_\star \in \{\zeta, \zeta + 1, \zeta^\theta, \zeta^\theta + 1\}$$

be the field element corresponding to $h_\star = h(\lambda_\star)$. By (8), we may choose $u_{\star,1} = U(\lambda_\star^{-\varrho}, 0)$, $u_{\star,2} = U(0, \lambda_\star^{\varrho+1})$, and $u_{\star,3} = U(\lambda_\star^{-\varrho}, \lambda_\star^{-(\varrho+1)})$. The field elements $\lambda_\star^{-\varrho}$, $\lambda_\star^{\varrho+1}$, $\lambda_\star^{-(\varrho+1)}$ are computable from the standard name by repeated squaring and inversion in $\mathbb{F}_q$, hence in $\mathsf{DTIME}(\mathrm{polylog}(n))$. Once these elements are computed as pairs $U(A,B) \in U$ we write them using $\mathrm{Word}_{A,B}$ (Lemma 6.4). Each such word has length $O(e) = O(\log q)$ and is computable in $\mathsf{DTIME}(\mathrm{polylog}(n))$.

Thus relation (17) is output as

$$h_\star t = \mathrm{Word}_{\lambda_\star^{-\varrho},0} \left(\mathrm{Word}_{0,\lambda_\star^{\varrho+1}}\right)^t \mathrm{Word}_{\lambda_\star^{-\varrho},\lambda_\star^{-(\varrho+1)}}.$$

Therefore relations (16) and (17) can be output uniformly in $\mathsf{DTIME}(\mathrm{polylog}(n))$. □

# 7 Conclusion and Open Questions

**Upper Bounds.** We placed Group Isomorphism into the second level of the polylogarithmic-time hierarchy (Theorem 3.1). As a consequence, Group Isomorphism can be solved by uniform $\mathsf{AC}$ circuits of depth-3, size $n^{\mathrm{polylog}(n)}$, and polylogarithmic fan-in at the bottom level (i. e., depth $2\frac{1}{2}$). This improved the depth over the previous-best: Collins, Grochow, Levet, and Weiß [CGLW25] showed that the more general Quasigroup Isomorphism problem belongs to the third level of the polylogarithmic time hierarchy, and hence can be solved by uniform $\mathsf{AC}$ circuits of depth-$3\frac{1}{2}$ and size $n^{O(\log n)}$.

Extending Theorem 3.1 to Quasigroup Isomorphism seems out of reach. Our technique depends heavily on both the theory of composition series for groups and the Classification of Finite Simple Groups. Classifying all finite simple quasigroups is considered hopeless, and for general quasigroups there is essentially no analogous theory of composition series that can be used the way we use it in this paper. It is even open whether all finite simple quasigroups are $O(1)$-generated. We

also point out that our approach of using polylogarithmically short presentations for finite (simple) groups unconditionally does not extend to the setting of quasigroups (Observation 1.4).

Assuming the Uniform Short Presentation Conjecture, we get even closer to depth 2 (see "Closer to depth 2..." on p. 4). We note that the Short Presentation Conjecture has been claimed to be solved, and even presented in talks, but not yet published (for more details see "The status of the (Uniform) Short Presentation Conjecture" on p. 4).

As part of our depth-$2\frac{1}{2}$ upper bound we also show that short presentations of $\mathrm{PSU}_3(q)$ and the Suzuki groups from the literature [HS01, GKKL08] can be made uniform (Propositions 5.1 and 6.1, respectively.) Although a short presentation for the Ree groups ${}^2G_2(q)$ is not yet available in the literature, in getting around this we develop tools for uniformity analogous to those we used for $\mathrm{PSU}_3(q)$ and the Suzuki groups. Namely, for all three groups we show that, relative to a certain generating set and in a certain coordinate-model of the groups, constructive membership in the Borel subgroup (or its unipotent radical) can be solved in $\mathsf{DTIME}(\mathrm{polylog}(n))$ (Lemma 3.5 for the Ree groups ${}^2G_2(q)$; Lemma 5.4 for ${}^2A_2(q) = \mathrm{PSU}_3(q)$; and Lemma 6.4 for the Suzuki groups ${}^2B_2(q) = \mathrm{Sz}(q)$.) Although the details of the generating sets and coordinate models differ from the three groups, we note the similarities between these lemmata (and their proofs).

**Lower bounds.** We also exhibit the first non-trivial circuit lower-bounds against Group Isomorphism. Precisely, we show that verifying that a multiplication table specifies a group, or even a Latin square (=quasigroup), requires DNFs of exponential size (Theorem 4.2). Furthermore, we show that isomorphism testing of Abelian groups requires CNFs of size $n^{\Omega(\log n)}$ (Theorem 4.3). The lower bound for CNFs relies crucially on a result of Seguins Pazzis [dSP10] bounding the dimension of an affine linear space of matrices of bounded rank. At the end of Section 4.2 we also discuss obstacles to extending our technique from matrices to tensors in attempting to get an $n^{\Omega(\log^2 n)}$ lower bound.

Our lower-bounds also hold more generally for Latin Square Isotopy (see Remark 1.1 and Corollary 1.2). Collins, Grochow, Levet, and Weiß [CGLW25] previously exhibited the same upper-bound for Latin Square Isotopy as for Quasigroup Isomorphism. Precisely, they showed that Latin Square Isotopy belongs to the third level of the polylogarithmic time hierarchy, and hence can be solved by uniform $\mathsf{AC}$ circuits of depth-$3\frac{1}{2}$ and size $n^{O(\log n)}$.

Circuit lower bounds are commonly obtained via either low-depth reductions from Parity or Majority, or via a switching lemma-type argument. The $\exists^{\log^2 n}\mathsf{FOLL}$ bound for Quasigroup Isomorphism due to Chattopadhyay, Torán, and Wagner [CTW13] unconditionally rules out the use of low-depth reductions from Parity or Majority. In particular, $\exists^{\log^2 n}\mathsf{FOLL}$ does not contain either Parity or Majority. Note that neither previous works [CTW13, CGLW25] nor our work here rules out the use of switching lemmas more generally. It might be possible to obtain circuit lower-bounds for other algebraic problems in the multiplication table model, through the use of low-depth reductions from validity checking (Theorem 4.2) or isomorphism testing of Abelian groups (Theorem 4.3).

**Open questions.** Perhaps the biggest question left open by our work is still to prove that Group Isomorphism is not in $\mathsf{AC}^0$ (or is!). As more approachable next steps, we highlight the following questions:

**Question 7.1.** Does Group Isomorphism require CNFs of size $n^{\Omega(\log^2 n)}$? Or more strongly of exponential size?

**Question 7.2.** Do depth-3 circuits for Group Isomorphism require super-polynomial size?

Because of our polynomial-size CNF for validity testing of multiplication tables (Lemma 3.3), the above two questions are equivalent to their promise versions.

**Question 7.3.** Does isomorphism testing of Abelian groups, in the multiplication table model (with the promise that the multiplication tables are valid multiplication tables for groups), require DNFs of super-polynomial size?

## Acknowledgments

M.L. thanks Martin Kassabov and James B. Wilson for helpful discussions. J. G. thanks Lance Fortnow for helpful discussions on how to name and discuss our low-level circuit classes.

G. K. is supported by the Templeton World Charity Foundation, Inc. (funder DOI: 501100011730) under the grant DOI:10.54224/20650 no. 20650 on "Building Diverse Intelligences through Compositionality and Mechanism Design", and additionally by the J. Grochow and R. Frongillo start-up funds at the University of Colorado Boulder. M. L. was partially supported by a Faculty Research and Development Grant from the College of Charleston. J. G. was supported by NSF CAREER grant CCF-2047756.

## LLM Usage Statement

The authors declare that no large language models (LLMs) were used in the research, writing, or editing of this work.

## A Mathematica code to verify matrix calculations symbolically in ${}^2G_2(q)$

We executed the following code using Wolfram 14.3.0.0 [WR26] on a MAC OS X ARM (64-bit) platform on 20 Aug 2026.

We use the function `AntidiagonalMatrix` by Sander Huisman [Hui19]:

```
ClearAll[AntidiagonalMatrix]
AntidiagonalMatrix[list : (_List | _SparseArray)] :=
 AntidiagonalMatrix[list, 0]
AntidiagonalMatrix[list : (_List | _SparseArray), k_Integer] :=
 Reverse[DiagonalMatrix[list, k], 2]
AntidiagonalMatrix[list : (_List | _SparseArray), k_Integer,
  n_Integer] := AntidiagonalMatrix[list, k, {n, n}]
AntidiagonalMatrix[list : (_List | _SparseArray), k_Integer,
  mn : {m_Integer, n_Integer}] :=
 Reverse[DiagonalMatrix[list, k, mn], 2]
```

We then set up our generating set of ${}^2G_2(q)$ from Lemma 3.5:

```
AlphaMat := Function[a, {{1, a^L, 0, 0, -a^(3*L + 1), -a^(3*L + 2), a^(4*L + 2) },
                         {0, 1, a, a^(L + 1), -a^(2*L + 1), 0, -a^(3*L + 2)},
                         {0, 0, 1, a^L, -a^(2*L), 0, a^(3*L + 1)},
                         {0, 0, 0, 1, a^L, 0, 0},
                         {0, 0, 0, 0, 1, -a, a^(L + 1)},
                         {0, 0, 0, 0, 0, 1, -a^L},
                         {0, 0, 0, 0, 0, 0, 1}}]
BetaMat := Function[b, {{1 , 0, -b^L, 0, -b, 0, -b^(L + 1)},
                        {0, 1, 0, b^L, 0, -b^(2*L), 0},
                        {0, 0, 1, 0, 0, 0, b},
                        {0, 0, 0, 1, 0, b^L, 0},
                        {0, 0, 0, 0, 1, 0, b^L},
                        {0, 0, 0, 0, 0, 1, 0},
                        {0, 0, 0, 0, 0, 0, 1}}]
GammaMat := Function[c, {{1 , 0, 0, -c^L, 0, -c, -c^(2*L)},
                         {0, 1, 0, 0, -c^L, 0, c},
                         {0, 0, 1, 0, 0, c^L, 0},
                         {0, 0, 0, 1, 0, 0, -c^L},
                         {0, 0, 0, 0, 1, 0, 0},
                         {0, 0, 0, 0, 0, 1, 0},
                         {0, 0, 0, 0, 0, 0, 1}}]
HMat := Function[lam,
  DiagonalMatrix[{lam^L, lam^(1 - L), lam^(2*L - 1), 1,
                  lam^(1 - 2*L), lam^(L - 1), lam^(-L)}]]
TMat := AntidiagonalMatrix[{-1, -1, -1, -1, -1, -1, -1}]
```

Here we left `L` (our $\ell$) as a symbol with no constraints. Although some expressions might be able to be simplified by taking into account the true semantics of $\ell$ in the context of the Ree groups, we turned out not to need that, as the elements we wanted appeared even with leaving `L` as an unconstrained variable.

Finally, we verify the needed calculations:

```
Part[HMat[lam] . AlphaMat[a] . BetaMat[b] . GammaMat[c] . TMat .
  AlphaMat[x] . BetaMat[y] . GammaMat[z], 7, 1]

> -lam^(-L)

Part[AlphaMat[a] . BetaMat[b] . GammaMat[c] . TMat . AlphaMat[x] .
  BetaMat[y] . GammaMat[z] , 6, 1]

> a^L

Part[BetaMat[b] . GammaMat[c] . TMat . AlphaMat[x] . BetaMat[y] . GammaMat[z],
     3, 1]

> -b
```

```
Part[GammaMat[c] . TMat . AlphaMat[x] . BetaMat[y] . GammaMat[z], 2, 1]

> -c
```

## References


[AD17] Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. *Inf. Comput.*, 256:2–8, 2017. Originally appeared in MFCS '14. doi:10.1016/j.ic.2017.04.004.

[Ajt83] M. Ajtai. $\Sigma^1_1$-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983. doi:10.1016/0168-0072(83)90038-6.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Ann. Math. (2)*, 160(2):781–793, 2004. doi:10.4007/annals.2004.160.781.

[Alb43] Abraham A. Albert. Quasigroups. I. *Trans. Amer. Math. Soc.*, 54(3):507–519, 1943. doi:10.2307/1990259.

[AMM77] Leonard M. Adleman, Kenneth L. Manders, and Gary L. Miller. On taking roots in finite fields. In *18th Annual Symposium on Foundations of Computer Science*, pages 175–178, 1977. doi:10.1109/SFCS.1977.18.

[AT11] Vikraman Arvind and Jacobo Torán. Solvable group isomorphism is (almost) in NP ∩ coNP. *ACM Trans. Comput. Theory*, 2(2):4:1–4:22, 2011. doi:10.1145/1944857.1944859.

[Bää14] Henrik Bäärnhielm. Recognising the small Ree groups in their natural representations. *J. Algebra*, 416:139–166, October 2014. doi:10.1016/j.jalgebra.2014.06.017.

[Bab16] László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In *STOC'16—Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 684–697. ACM, New York, 2016. Preprint of full version at arXiv:1512.03547v2 [cs.DS]. doi:10.1145/2897518.2897542.

[Bae38] Reinhold Baer. Groups with abelian central quotient group. *Transactions of the American Mathematical Society*, 44(3):357–386, 1938. doi:10.2307/1989886.

[Bar92] D.A.M. Barrington. Quasipolynomial size circuit classes. In *[1992] Proceedings of the Seventh Annual Structure in Complexity Theory Conference*, pages 86–93, 1992. doi:10.1109/SCT.1992.215383.

[BB05] Arnold Beckmann and Samuel R. Buss. Separation results for the size of constant-depth propositional proofs. *Ann. Pure Appl. Logic*, 136(1-2):30–55, 2005. doi:10.1016/j.apal.2005.05.002.

[BCGQ11] László Babai, Paolo Codenotti, Joshua A. Grochow, and Youming Qiao. Code equivalence and group isomorphism. In *Proceedings of the Twenty-Second Annual ACM–SIAM Symposium on Discrete Algorithms (SODA11)*, pages 1395–1408, Philadelphia, PA, 2011. SIAM. doi:10.1137/1.9781611973082.107.

[BDHM92] Gerhard Buntrock, Carsten Damm, Ulrich Hertrampf, and Christoph Meinel. Structure and importance of logspace-mod class. *Math. Syst. Theory*, 25(3):223–237, 1992. doi:10.1007/BF01374526.

[BDPW10] Paul Beame, Matei David, Toniann Pitassi, and Philipp Woelfel. Separating deterministic from randomized multiparty communication complexity. *Theory of Computing*, 6(1):201–225, 2010. doi:10.4086/toc.2010.v006a009.

[BE99] Hans Ulrich Besche and Bettina Eick. Construction of finite groups. *J. Symb. Comput.*, 27(4):387–404, 1999. doi:10.1006/jsco.1998.0258.

[BEO02] Hans Ulrich Besche, Bettina Eick, and E.A. O'Brien. A millennium project: Constructing small groups. *Intern. J. Alg. and Comput*, 12:623–644, 2002. doi:10.1142/S0218196702001115.

[BGK+97] László Babai, Albert J. Goodman, William M. Kantor, Eugene M. Luks, and Péter P. Pálfy. Short presentations for finite groups. *Journal of Algebra*, 194(1):79–112, 1997. doi:10.1006/jabr.1996.6980.

[BLS84] Ronald V. Book, Timothy J. Long, and Alan L. Selman. Quantitative relativizations of complexity classes. *SIAM Journal on Computing*, 13(3):461–487, 1984. doi:10.1137/0213030.

[BS84] László Babai and Endre Szemerédi. On the complexity of matrix group problems I. In *25th Annual Symposium on Foundations of Computer Science, West Palm Beach, Florida, USA, 24-26 October 1984*, pages 229–240. IEEE Computer Society, 1984. doi:10.1109/SFCS.1984.715919.

[BS20] Jendrik Brachter and Pascal Schweitzer. On the Weisfeiler–Leman dimension of finite groups. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 287–300. ACM, 2020. doi:10.1145/3373718.3394786.

[Bsh26] Nader H. Bshouty. Sublinear time algorithms for abelian group isomorphism and basis construction. In Jakub Kozik and Alexander Wolff, editors, *SOFSEM 2026: Theory and Practice of Computer Science - 51st International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2026, Kraków, Poland, February 9-13, 2026, Proceedings*, volume 16448 of *Lecture Notes in Computer Science*, pages 31–45. Springer, 2026. doi:10.1007/978-3-032-17801-5_3.

[Car89] Roger W Carter. *Simple groups of Lie type*, volume 22. John Wiley & Sons, New York, 1989.

[CFI92] Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992. Originally appeared in SFCS '89. doi:10.1007/BF01305232.

[CGLW25] Nathaniel A. Collins, Joshua A. Grochow, Michael Levet, and Armin Weiß. On the constant-depth circuit complexity of generating quasigroups. *TheoretiCS*, 4:1–39, 2025.

Originally appeared at ISSAC '24. Full version also available as arXiv:2402.00133. doi:10.46298/theoretics.25.19.

[CH03] John J. Cannon and Derek F. Holt. Automorphism group computation and isomorphism testing in finite groups. *J. Symb. Comput.*, 35:241–267, March 2003. doi:10.1016/S0747-7171(02)00133-5.

[CL24] Nathaniel A. Collins and Michael Levet. Count-free Weisfeiler–Leman and group isomorphism. *International Journal of Algebra and Computation*, 34(03):283–330, 2024. doi:10.1142/S0218196724500103.

[CM80] H. S. M. Coxeter and W. O. J. Moser. *Generators and relations for discrete groups. 4th ed.*, volume 14 of *Ergeb. Math. Grenzgeb.* Springer-Verlag, Berlin, 1980. doi:10.1007/978-3-662-21943-0.

[Col23] Nathaniel A. Collins. Weisfeiler–Leman and group isomorphism, 2023. Undergraduate Thesis. University of Coloardo Boulder. URL: https://scholar.colorado.edu/concern/undergraduate_honors_theses/6m311q80b.

[CTW13] Arkadev Chattopadhyay, Jacobo Torán, and Fabian Wagner. Graph isomorphism is not $AC^0$-reducible to group isomorphism. *ACM Trans. Comput. Theory*, 5(4):Art. 13, 13, 2013. Preliminary version appeared in FSTTCS '10. doi:10.1145/2540088.

[Cur65] Charles W. Curtis. Central extensions of groups of Lie type. *Journal für die reine und angewandte Mathematik*, 1965(220):174–185, 1965. doi:doi:10.1515/crll.1965.220.174.

[Ded64] Paul Dedecker. Les foncteurs $Ext_\Pi$, $\mathrm{H}_\Pi{}^2$ et $H_\Pi{}^2$ non abéliens. *C. R. Acad. Sci. Paris*, 258:4891–4894, 1964.

[Deh12] M. Dehn. Über unendliche diskontinuierliche Gruppen. *Math. Ann.*, 71:116–144, 1912. English translation in [Deh87]. URL: https://eudml.org/doc/158521, doi:10.1007/BF01456932.

[Deh87] Max Dehn. *Papers on group theory and topology. Translated and introduced by John Stillwell.* New York, NY: Springer, reprint of the hardback ed. edition, 1987. doi:10.1007/978-1-4612-4668-8.

[DF13] Rodney G. Downey and Michael R. Fellows. *Fundamentals of parameterized complexity.* Texts Comput. Sci. London: Springer, 2013. doi:10.1007/978-1-4471-5559-1.

[dSP10] Clément de Seguins Pazzis. The affine preservers of non-singular matrices. *Archiv der Mathematik*, 95:333–342, 2010. doi:10.1007/s00013-010-0173-6.

[DW22] Heiko Dietrich and James B. Wilson. Group isomorphism is nearly-linear time for most orders. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 457–467, 2022. doi:10.1109/FOCS52979.2021.00053.

[ELGO02] Bettina Eick, C. R. Leedham-Green, and E. A. O'Brien. Constructing automorphism groups of $p$-groups. *Comm. Algebra*, 30(5):2271–2295, 2002. doi:10.1081/AGB-120003468.

[FGSTT20] Flavio Ferrarotti, Senén González, Klaus-Dieter Schewe, and José María Turull-Torres. Proper hierarchies in polylogarithmic time and absence of complete problems. In Andreas Herzig and Juha Kontinen, editors, *Foundations of Information and Knowledge Systems (FoIKS)*, pages 90–105, Cham, 2020. Springer International Publishing. doi:10.1007/978-3-030-39951-1_6.

[FN70] V. Felsch and J. Neubüser. On a programme for the determination of the automorphism group of a finite group. In *Computational Problems in Abstract Algebra (Proc. Conf., Oxford, 1967)*, pages 59–60. Pergamon Press, 1970.

[FSS84] Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Math. Syst. Theory*, 17(1):13–27, 1984. doi:10.1007/BF01744431.

[GG15] George Glauberman and Lukasz Grabowski. Groups with identical $k$-profiles. *Theory Comput.*, 11:395–401, 2015. doi:10.4086/TOC.2015.V011A015.

[GJL25] Joshua A. Grochow, Dan Johnson, and Michael Levet. Complexity of identifying Fitting-free groups. In Artur Jez and Jan Otop, editors, *Fundamentals of Computation Theory - 25th International Symposium, FCT 2025, Wrocław, Poland, September 15-17, 2025, Proceedings*, volume 16106 of *Lecture Notes in Computer Science*, pages 208–220. Springer, 2025. Preprint of full version at arXiv:2504.19777 [cs.CC]. doi:10.1007/978-3-032-04700-7_16.

[GKKL08] R. M. Guralnick, W. M. Kantor, M. Kassabov, and A. Lubotzky. Presentations of finite simple groups: a quantitative approach. *J. Am. Math. Soc.*, 21(3):711–774, 2008. Preprint arXiv:math/0602508 [math.GR]. doi:10.1090/S0894-0347-08-00590-0.

[GL26] Joshua A. Grochow and Michael Levet. On the parallel complexity of group isomorphism via Weisfeiler–Leman. *Journal of Computer and System Sciences*, 156:103703, 2026. Preliminary version appeared in the proceedings of FCT 2023, DOI:10.1007/978-3-031-43587-4_17. doi:10.1016/j.jcss.2025.103703.

[Gow11] Tim Gowers. Comment on Dick Lipton's blog entry: The group isomorphism problem: A possible polymath problem?, Nov 2011. URL: https://rjlipton.com/2011/11/07/the-group-isomorphism-problem-a-possible-polymath-p

[GQ17] Joshua A. Grochow and Youming Qiao. Algorithms for group isomorphism via group extensions and cohomology. *SIAM J. Comput.*, 46(4):1153–1216, 2017. Preliminary version in IEEE Conference on Computational Complexity (CCC) 2014 (DOI:10.1109/CCC.2014.19). Also available as arXiv:1309.1776 [cs.DS] and ECCC Technical Report TR13-123. doi:10.1137/15M1009767.

[Hig60] Graham Higman. Enumerating $p$-groups. I: Inequalities. *Proc. Lond. Math. Soc. (3)*, 10:24–30, 1960. doi:10.1112/plms/s3-10.1.24.

[HKSW20] Alexander Hulpke, Martin Kassabov, Akos Seress, and James B. Wilson. Presentations of simple groups of Ree type. Unpublished. See videos Kassabov presentation at the Newton Institute (March 2020) and

Kassabov presentation at the Hausdorff Research Institute for Mathematics (August 2025)., 2020.

[HS01] Alexander Hulpke and Ákos Seress. Short presentations for three-dimensional unitary groups. *J. Algebra*, 245(2):719–729, 2001. doi:10.1006/jabr.2001.8943.

[Hui19] Sander Huisman. AntidiagonalMatrix. https://resources.wolframcloud.com/FunctionRepositor in the Wolfram Function Repository, 2019.

[HvdH21] David Harvey and Joris van der Hoeven. Integer multiplication in time $O(n \log n)$. *Ann. Math. (2)*, 193(2):563–617, 2021. doi:10.4007/annals.2021.193.2.4.

[Hå86] J. Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 6–20, New York, NY, USA, 1986. Association for Computing Machinery. doi:10.1145/12130.12132.

[Ina97] H. Inassaridze. Non-abelian cohomology of groups. *Georgian Math. J.*, 4(4):313–332, 1997. doi:10.1023/A:1022938428031.

[JLVW26] Dan Johnson, Michael Levet, Petr Vojtěchovský, and Brett Widholm. Parallel complexity of identifying groups and quasigroups via decompositions. In Uli Fahrenberg, Wesley Fussner, and Luigi Santocanale, editors, *Relational and Algebraic Methods in Computer Science*, pages 188–207, Cham, 2026. Springer Nature Switzerland. Preprint of full version at arXiv:2508.06478 [cs.DS]. doi:10.1007/978-3-032-22469-9_11.

[KLM01] Gregor Kemper, Frank Lübeck, and Kay Magaard. Matrix generators for the Ree groups ${}^2G_2(q)$. *Commun. Algebra*, 29(1):407–413, 2001. doi:10.1081/AGB-100000808.

[Lev23] Michael Levet. On the complexity of identifying strongly regular graphs. *Australasian J. Combin.*, 87:41–67, 2023. URL: https://ajc.maths.uq.edu.au/pdf/87/ajc_v87_p041.pdf.

[Lev26] Michael Levet. Parallel Algorithms for Group Isomorphism via Code Equivalence. In Pierre Fraigniaud, editor, *20th Scandinavian Symposium on Algorithm Theory (SWAT 2026)*, volume 370 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 30:1–30:19, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. Preprint of full version at arXiv:2604.13953 [cs.CC]. doi:10.4230/LIPIcs.SWAT.2026.30.

[LGR16] François Le Gall and David J. Rosenbaum. On the group and color isomorphism problems. arXiv:1609.08253 [cs.CC], 2016.

[LRS24] Michael Levet, Puck Rombach, and Nicholas Sieger. Canonizing Graphs of Bounded Rank-Width in Parallel via Weisfeiler-Leman. In Hans L. Bodlaender, editor, *19th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2024)*, volume 294 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:18, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SWAT.2024.32.

[LSZ77] R. J. Lipton, L. Snyder, and Y. Zalcstein. The complexity of word and isomorphism problems for finite groups. Yale University Dept. of Computer Science Research Report # 91, 1977. URL: https://apps.dtic.mil/dtic/tr/fulltext/u2/a053246.pdf.

[Luk15] Eugene M. Luks. Group isomorphism with fixed subnormal chains, 2015. arXiv:1511.00151.

[Mes85] Roy Meshulam. On the maximal rank in a subspace of matrices. *The Quarterly Journal of Mathematics*, 36(2):225–229, 1985. doi:10.1093/qmath/36.2.225.

[Mil78] Gary L. Miller. On the $n^{\log n}$ isomorphism technique (a preliminary report). In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, STOC '78, pages 51–58, New York, NY, USA, 1978. Association for Computing Machinery. doi:10.1145/800133.804331.

[NT17] André Nies and Katrin Tent. Describing finite groups by short first-order sentences. *Isr. J. Math.*, 221(1):85–115, 2017. Preprint arXiv:1409.8390. doi:10.1007/s11856-017-1563-2.

[Pyb93] L. Pyber. Enumerating finite groups of given order. *Ann. Math. (2)*, 137(1):203–220, 1993. doi:10.2307/2946623.

[QST11] Youming Qiao, Jayalal M. N. Sarma, and Bangsheng Tang. On isomorphism testing of groups with normal Hall subgroups. In *Proc. 28th STACS*, pages 567–578, 2011. doi:10.4230/LIPIcs.STACS.2011.567.

[Rab80] Michael O. Rabin. Probabilistic algorithms in finite fields. *SIAM J. Comput.*, 9:273–280, 1980. doi:10.1137/0209024.

[Ree60] Rimhak Ree. A family of simple groups associated with the simple Lie algebra of type $(G_2)$. *Bull. Am. Math. Soc.*, 66:508–510, 1960. doi:10.1090/S0002-9904-1960-10523-X.

[Rob82] Derek J.S. Robinson. *A Course in the Theory of Groups*. Springer, 1982. doi:https://doi.org/10.1007/978-1-4419-8594-1.

[Ros13] David J. Rosenbaum. Bidirectional collision detection and faster deterministic isomorphism testing. arXiv:1304.3935 [cs.DS], 2013.

[Sel94] Alan L. Selman. A taxonomy of complexity classes of functions. *J. Comput. Syst. Sci.*, 48(2):357–381, 1994. doi:10.1016/S0022-0000(05)80009-1.

[Sim65] Charles C. Sims. Enumerating $p$-groups. *Proceedings of the London Mathematical Society*, s3-15(1):151–166, 01 1965. doi:10.1112/plms/s3-15.1.151.

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC), 1987, New York, New York, USA*, pages 77–82. ACM, 1987. doi:10.1145/28395.28404.

[Suz62] Michio Suzuki. On a class of doubly transitive groups. *Ann. Math. (2)*, 75:105–145, 1962. URL: hdl.handle.net/2027/mdp.39015095249804, doi:10.2307/1970423.

[Tan13] Bangsheng Tang. *Towards Understanding Satisfiability, Group Isomorphism and Their Connections*. PhD thesis, Tsinghua University, 2013.

[Tit74] Jacques Tits. *Buildings of Spherical Type and Finite BN-Pairs*. Springer Berlin, Heidelberg, 1 edition, 1974. doi:10.1007/978-3-540-38349-9.

[Tor04] Jacobo Torán. On the hardness of graph isomorphism. *SIAM J. Comput.*, 33(5):1093–1108, 2004. doi:10.1137/S009753970241096X.

[vLW01] J. H. van Lint and R. M. Wilson. *A course in combinatorics*. Cambridge: Cambridge University Press, 2nd edition, 2001. doi:10.1017/CBO9780511987045.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity - A Uniform Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1999. doi:10.1007/978-3-662-03927-4.

[Wag10] F. Wagner. *On the complexity of isomorphism testing for restricted classes of graphs*. PhD thesis, Universität Ulm, 2010. URL: https://oparu.uni-ulm.de/xmlui/bitstream/handle/123456789/3923/vts_7264_10267.pdf.

[War66] Harold N. Ward. On Ree's series of simple groups. *Trans. Amer. Math. Soc.*, 121(1):62, January 1966. doi:10.2307/1994333.

[Wil19] James B. Wilson. The threshold for subgroup profiles to agree is logarithmic. *Theory of Computing*, 15(19):1–25, 2019. doi:10.4086/toc.2019.v015a019.

[Wol94] Marty J. Wolf. Nondeterministic circuits, space complexity and quasigroups. *Theoretical Computer Science*, 125(2):295–313, 1994. doi:10.1016/0304-3975(92)00014-I.

[WR26] Inc. Wolfram Research. Mathematica, Version 14.3.0.0, 2026. Champaign, IL. URL: https://www.wolfram.com/mathematica.

[Yao85] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, SFCS '85, pages 1–10, USA, 1985. IEEE Computer Society. doi:10.1109/SFCS.1985.49.

[ZKT85] V. N. Zemlyachenko, N. M. Korneenko, and R. I. Tyshkevich. Graph isomorphism problem. *J. Soviet Math.*, 29(4):1426–1481, May 1985. doi:10.1007/BF02104746.



ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il